TURING

产品经理面试宝典

Cracking the PM Interview

How to Land a Product Manager Job in Technology

【美】Gayle Laakmann McDowell Jackie Bavaro 著
吴海星 陈少芸 译



原谷歌资深面试官经验之作，从简历和面试注意事项到职业发展规划
一步步指导应届毕业生及工程师应聘产品经理


中国工信出版集团


人民邮电出版社
POSTS & TELECOM PRESS

数字版权声明



Gayle Laakmann McDowell

美国求职咨询网站CareerCup.com创始人兼CEO，著有《程序员面试金典》和《金领简历：敲开苹果、微软、谷歌的大门》。曾供职于谷歌、微软和苹果等公司，是谷歌招聘委员会的成员。拥有宾州大学计算机科学的学士学位和硕士学位，以及沃顿商学院的MBA学位。



Jackie Bavaro

初创公司Asana的产品经理，为Dropbox、Airbnb、Uber、Foursquare和Pinterest等公司构建现代办公软件。曾任谷歌产品经理，微软规划经理。拥有康奈尔大学计算机科学与经济学双学位。



Cracking the PM Interview: How to Land a Product Manager Job in Technology

产品经理面试宝典

[美] Gayle Laakmann McDowell　Jackie Bavaro　著

吴海星　陈少芸　译

人民邮电出版社

北　京

图书在版编目（CIP）数据

产品经理面试宝典 / (美) 麦克道尔
(McDowell,G.L.)，(美) 巴瓦罗 (Bavaro,J.) 著 ; 吴
海星, 陈少芸译. -- 北京 : 人民邮电出版社, 2015.3
ISBN 978-7-115-38390-7

Ⅰ. ①产… Ⅱ. ①麦… ②巴… ③吴… ④陈… Ⅲ.
①IT产业一产品管理 Ⅳ. ①F49

中国版本图书馆CIP数据核字(2015)第017762号

内 容 提 要

本书针对 IT 行业产品经理，以面试为主线，首先介绍产品经理职责以及谷歌、微软等知名企业中产品经理的作用和要求；然后采访了几位知名企业的产品经理，介绍成为产品经理的基本素质；之后从简历准备、各公司面试要点到具体面试问题进行详细分析，这部分是本书的重点内容。

读者对象包括 IT 行业产品经理以及对如何做好产品有兴趣的人士。

◆ 著　[美] Gayle Laakmann McDowell，Jackie Bavaro
译　吴海星　陈少芸
责任编辑　朱　巍
执行编辑　杨　琳
责任印制　杨林杰

◆ 人民邮电出版社出版发行　北京市丰台区成寿寺路11号
邮编　100164　电子邮件　315@ptpress.com.cn
网址　http://www.ptpress.com.cn
北京　印刷

◆ 开本：800×1000　1/16
印张：17.5
字数：403千字　2015年 3 月第 1 版
印数：1 – 4 000册　2015年 3 月北京第 1 次印刷

著作权合同登记号　图字：01-2014-1315号

定价：59.00元

读者服务热线：(010)51095186转600　印装质量热线：(010)81055316

反盗版热线：(010)81055315

广告经营许可证：京崇工商广字第 0021 号

版 权 声 明

致　　谢

感谢所有在写书的过程中给予我支持的人。

Paul Unterberg、Adam Kazwell以及两位不愿透露姓名的人，我向你们致以深深的谢意，感谢你们愿意在这本书中展现自己的简历。

Fernando Delgado、Ashley Carroll、Brandon Bray、Thomas Arend、Johanna Wright、Lisa Kostova Ogata、Ian McAllister、Adam Nash、Sachin Rekhi和Ken Norton，我和Jackie以及我们的读者非常感激你们对此书的贡献。

那些提出建议及反馈的产品经理（及相关）朋友、同事们，也谢谢你们！你们棒极了！

感谢我的丈夫John和我的婆婆Donna，正是你们不间断的支持使得此书得以问世。

我那漂亮的儿子Davis，总有一天你会长大，会读到这一段话，到时你就会知道我有多爱你。

——盖尔

感谢爱意浓浓的家人。我的父亲对收费大桥上的罗嗦行文十分不满，我拿这个例子来作第一道产品经理面试题。我的丈夫在我写书的过程中不断鼓励我，我的猫也常伸爪相助。

非常感谢Steven Sinofsky和Marissa Mayer，是他们雇用我到了一支才华横溢的产品经理队伍，并且帮助提高了产品管理的水平。

非常感谢我那几位出色的经理——Mike Morton、Tom Stocky、Jack Menzel、Johanna Wright和Justin Rosenstein，你们是我了不起的导师和榜样。

感谢那些与我分享产品经理的经验和建议的人。很多人可谓英雄所见略同，这真让我吃惊。我要特别感谢下面这几个人：感谢Shirin Oskooi，你帮我迈出了第一步；感谢Daniel Dulitz，你提议写一个关于从设计师转型为产品经理的章节；感谢Nundu Janakiram，你提议我写一个关于产品经理误区的章节；感谢Chrix Finne，你为我介绍了许多很有帮助的人。

——杰基

目　录

第 1 章 简 介

产品经理是个奇怪的职位。

对很多职位而言，从 A 点到 B 点的路径相当清晰。如果你要做编程工作，就去学习如何编程。上大学读计算机科学专业，或者自学如何编程。做会计师、律师、医生等，无不如此。

如果想做产品经理，你该怎么办呢？没有教产品管理的学校。没有正规的培训。一般也没有对应的提升职位。

那么，如何才能当上产品经理呢？本书就要来讲讲这个问题。

杰基我们俩跟不计其数的产品经理都有过合作，帮他们学习如何积累合适的经验，如何摆正自己的位置，如何准备面试，以及如何轻松拿下这一职位。

花费了大量时间的训练课程和对话，最终转换成了这本书。

1.1 这个为什么重要

产品管理不应该是这么虚幻的职位，只有那些足够幸运（并且有关系）的人，才能听别人解释产品管理究竟是什么，才能涉足这个岗位。其实，有更多的人参与进来，对应聘者和雇主都是好事情。

作为应聘者，你可以更好地根据职位需要调整自己。不是说要让你去装，而是只有知道自己缺什么，才能真正去获取那些经验。

当要面试时，你可以有效地针对他们的问题作准备。你要知道如何介绍自己独特的经验，以及最重要的工作成果；还要学会如何解答面试问题；而且要深刻理解从用户出发意味着什么。最终还能夯实你的专业技能。

反过来说，雇主也能找到更多胜任的应聘者。放松而准备充分的应聘者比较能表现出自己真

实的水平。他们知道哪些工作成果最能引起雇主的兴趣，最能说明问题，并且能将对话向那些事情上引导。只要真正理解问题的指向，他们就能展示出解决问题的高超技能。他们可以学会应对特定问题所需的知识点。这种准备过程可以考验应聘者从事实际工作的技能。

把面试过程中的猜测和盲目性去掉，对所有人都是好事。

1.2　我们是谁

从作者介绍中你可以看到我们的经历：谷歌、微软、苹果、创业公司、招聘委员会、大量的面试和教学课程，等等。

这一切都很不错，但还不能让你了解我们真正的身份，以及我们为什么要写这样一本书讲授如何得到产品经理的职位。

我（盖尔）有着深厚的工程背景，但我也花了很多时间来做跟招聘有关的工作：招聘开发者和产品经理，进行模拟面试，教应聘者如何改善自己的答案，教会他们不懂的概念，发现他们的愿景和激情。

通过这些我学到两件事情。首先，我认识到，即便应聘者很优秀，但只要稍加指导，他们在面试中的表现也有很大的提升空间；其次，我发现有关应聘产品经理职位的信息非常少。很多人都在谈论如何做一个优秀的产品经理，但很少有人谈论如何才能进入这个领域。

当然，除了我那位迷人的合作者杰基。

我在问答网站 Quora 上无意中发现了她的博客，就关注上了。过了一段时间，有几件事打动了我。首先，她在几家顶级公司工作过，所以她自己就可以把这些面试吃透；第二，她的建议切中要点，不但可行而且非常优秀；第三，她重视。她足够重视，所以写博客来帮人们进入产品经理这个领域。

让更多的人可以涉足产品管理领域对我和杰基来说都很重要。我们相信每个人都可以从中受益，因此我们就写了这本书。

1.3　现在怎么办

这本书不是让你假装成能够胜任产品经理的工作，而是要给你一个有始有终的作战计划。

我们会告诉你产品经理是什么样的，这个职位在不同的公司会有什么样的变化。我们还会介绍不同公司是如何考虑他们的招聘流程的。

我们还会跟各种背景的产品经理交谈，了解他们是如何进入这一领域的，看看他们认为优秀

的产品经理应具备什么样的特质。我们还要探讨产品经理一般会有哪些背景，你又如何根据自己的背景和兴趣作出这种转变。

我们会告诉你如何写出出色的简历和求职信。其中有些建议不局限于产品经理，但大部分是专门针对产品经理的。

我们会建议你如何准备面试：面试前需要做哪些研究工作，为了回答问题还要作哪些准备。我们会分析几类主要的面试问题，主要包括：行为问题、评估问题、产品问题、案例问题和编码问题。而且，我们会向你解释面试官想知道什么，以及你应该如何掌握这些问题。

我们会帮你展示出自己最好的一面，帮你成功入职。

在阅读这些内容时，你可以想想如何在面试时亮出你的优势，如何做出强有力的响应。

我们是来帮你的。请加入我们的网站crackingthepminterview.com获取更多资源和建议。

最后，就像软件永远不会“做完”一样，我们希望这本书也不会。如果你有建议、反馈、问题或者只想打个招呼，不要犹豫，给我们发邮件吧：gayleandjackie@careercup.com

谢谢，祝你好运！

盖尔·L. 麦克道尔
facebook.com/gayle
twitter.com/gayle
technologywoman.com
quora.com/Gayle-Laakmann-McDowell

杰基·巴瓦罗
facebook.com/jackie.bavaro
twitter.com/jackiebo
pmblog.quora.com
quora.com/Jackie-Bavaro

第 2 章 产品经理的职责

2.1 什么是产品经理

产品经理（PM，Product Manager）的职责是确保团队能交出伟大的产品。

有人说产品经理（有时也叫专案经理或项目经理）就像产品的小 CEO。从某些角度来说这完全正确，因为产品经理要全面承担产品的责任，事无巨细面面俱到。产品经理要确立愿景和策略，产品经理要定义成功的标准，产品经理要作出决定。

但把产品经理说成 CEO 漏了最重要的一点：产品经理没有团队的直接人事权。

作为产品经理，你需要学会如何在没有权力的情况下领导你的团队，用你的愿景和研究影响他们。产品经理在大多数公司里都非常受尊敬，但工程师更受尊敬。如果你摆架子，对身边的人颐指气使，就会发现可能很难把事情做成。毕竟工程师才是真正构建产品的人。你需要得到他们的支持。

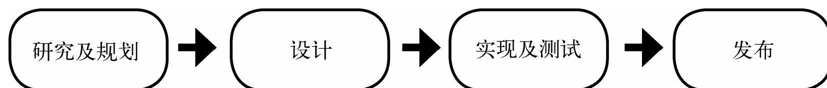
产品管理这一职业之所以这么吸引人，其中一个原因是它综合了技术、业务和设计三方面的知识和技能。你要扮演很多角色，并且要了解多种观点。

作为产品经理，你要为客户代言。你得了解他们的需求，并将这些需求转换成产品的目标和特性。然后你还得确保以一种内聚的、精心设计的方式构建那些特性，能真正解决客户的需求。从总体大局到细枝末节，一切你都要关注。你可能头一天还在对团队的三年愿景做头脑风暴，而第二天就得处理对话框按钮的各种细节。

产品管理是一个需要高度协作的岗位。产品经理通常是工程师和其他角色之间的主要联络人，包括但不限于设计、质保、用户调研、数据分析、营销、销售、客户支持、业务开发、法务、文案、其他工程团队，以及管理团队。产品经理通常要负责确定何时引入哪个团队，如果没有相应的团队，还要负责填补那个空白。

2.2 产品经理的职能

产品经理的日常工作会随着产品生命周期的过程发生变化。一开始，你得确定要构建什么；中间要帮团队取得进展；最后要准备推出产品。



尽管不同公司的产品生命周期是不同的（有时甚至不同的团队都不一样），但通常都会遵循研究及规划、设计、实现及测试、发布这样一个通用的模式。当然，这些步骤经常是相互叠加的，并且彼此之间相互反馈。

有些公司或团队将产品经理的职责分给两个人：一个偏重于业务，一个偏重于工程。在采用这种划分方式的公司中，偏重工程的那个人被称为技术专案经理或技术产品经理（TPM），偏重于业务那个人被称为产品经理（PM）。

如果团队中有一个 TPM 和一个 PM，则产品经理专注于研究规划和发布，而技术产品经理更多的是专注于设计以及实现和测试。比如说，产品经理会研究市场并定义需求。TPM 会跟 PM 一起合作，将那些需求变成要完成的特定功能的工作，然后促成工程团队构建它。

2.2.1 研究及规划

所有的产品和功能都是从研究及规划开始的。这时候产品经理开始思考接下来该做什么。下一个点子可能来自于客户的需求、竞争分析、新技术、用户研究、销售或营销团队、头脑风暴，或者产品的总体愿景。

根据该职责的范围，产品经理在这一阶段的大部分工作是创建或提出一个路线图。也就是说要给团队找出一个有凝聚力的长期规划。产品经理要跟所有可能的信息源交谈，列出潜在特性或开发工作的大清单。然后根据客户需要、竞争格局、业务需要和团队的专业知识等因素，确定各种功能和方案的优先级。

产品经理提出路线图后，就要争取其他人的认同。在某些公司，比如微软、苹果和亚马逊，有一个自顶向下的审批过程，高管和董事们很早就会参与其中。而其他公司，比如谷歌、Facebook 和很多创业公司，采用自底向上的途径，产品经理的工作重点是争取工程师的认可。

他一旦选定特性集，就意味着产品经理成了这方面的专家。他要深入思考待解决的问题，想清楚那些功能的目的。在后续阶段，团队中的所有人都会提出各种问题，包括“我们为什么要做这个”，那时候就需要产品经理给出答案。

这也是产品经理开始定义成功标准的时候。他会想象团队成功后的世界是什么样的。很多公

司用目标和关键结果（OKR）模型来交流团队最重要的目标。在这个模型中，产品经理跟团队一起拿出他们能够为之努力的可测量结果。

2.2.2　设计

一旦产品经理与大家达成一致，知道团队将要构建什么，就该设计产品和特性了。

产品设计不只是用户界面（UI）设计，也不仅是要勾勒出产品看起来是什么样的。产品设计是要定义产品的特性和功能。在不同的公司和团队中，产品经理在产品设计中所扮演的角色会有很大变化。

在某些团队里，特别是在微软发运软件（跟在线软件相对）的团队中，产品经理要写出详尽的功能规范（spec），其中包括：

- 目标
- 用例
- 需求
- 框架图
- 描述该功能的每一个可能状态的要点
- 国际化
- 安全性

然后会由开发人员、测试人员和其他产品经理用几周的时间对这个规范进行检验、评审和迭代。在这些团队里，所有面向用户的决定都要由产品经理来作。

其他团队的规范要松散得多，并且设计过程也更迅速。产品经理可能就是跟一位交互设计师坐一坐，聊聊目标，在白板上头脑风暴一下，然后在设计师的模拟产品上给出反馈，不断进行迭代。模拟产品准备好后，产品经理可能会把它们发给工程师，邮件里可能只写几句话。在这些团队中，一般由工程师作出比较容易的产品决策，再去跟产品经理商量那些困难的决定。

对某些团队来说，特别是在苹果，设计工作大部分是由专门的设计团队完成的，基本不需要产品经理参与。在那种团队里，产品经理可能更像是项目管理和救火队员的角色。

因为产品经理在产品设计中的作用可能变化很大，所以在你面试时一定要问清楚。问问你的核心团队和外围团队都有谁一起共事。找出你要用多长时间写规范，以及要跟设计师合作多久。了解清楚在形成产品决策时，产品经理、设计师和工程师能在哪里取得平衡。

2.2.3　实现及测试

并不是工程师一开始编码就算产品经理完成任务了。在实现阶段，产品经理还要跟踪项目的

进展情况，并及时作出调整。

在实现过程中，产品经理最重要的一项任务是帮工程师有效地工作。产品经理会定期检查他的团队，了解工作进展。

工程师经常会因为等待其他团队的工作成果而被卡住。这时产品经理要给工程师安排其他任务，同时跟那个团队配合，促成瓶颈任务尽快结束。

有时会发现实现一个功能的困难程度超出了预期，此时产品经理要想办法修改这样的功能，让它变得更容易实现。如果某个工程师的进度拖后了，产品经理可以认真地重新审查这项工作的进度安排，并砍掉优先级较低的工作。

在实现过程中，产品经理还要开始收集早期产品的反馈并报告 bug。有时在设计阶段看起来很好的特性，在现实世界中可能达不到预期那么好的效果。为了找出这样的问题，团队要做可用性研究，试运行，并且在内部做“自我体验”（dogfooding）。

自我体验源自“自己酿的酒自己喝”，就是使用自己产品的意思。比如微软的人一直都会提前在他们的电脑上运行下一版的 Windows，Facebook 的员工也用 Facebook 群组交流。

有时候为了找出自己产品的使用办法，需要团队更有创造性。比如谷歌会给员工划拨 AdWords 预算，鼓励他们创造广告营销活动，以确保他们对自己的产品有充分的体验。

可用性研究是在产品推出前检验它是否好用的另一种办法。在可用性研究中，参与者会尝试一个新产品或特性的早期原型。这些参与者通常会得到一个情景或目标，并试着用这个原型完成目标。

比较大的公司一般会有专职的用户研究员，根据产品经理给出的输入条件开展研究。比较小的公司可能是产品经理做这些研究。不论哪种情况，产品经理都可以通过观察这些研究来判断困扰人们的地方，并找出关键的可用性问题。

自我体验和可用性研究对收集定性的反馈很有利，而如果你的软件是在线的，试运行可以帮你得到定量反馈。在试运行阶段，新特性会开放给一部分用户（实验组），而其他用户（控制组）仍然使用不带新特性的产品。在线软件的新特性通常都是凭借着这样的实验逐渐推出的。

你可以在实验中测量新特性的特定指标，比如有多少用户点击了你新加上去的按钮，以及像用户参与度、留存比例和收益等多项总体成功指标。通过比较实验组和控制组的成功指标，你可以判断出新特性究竟有多成功。

有了自我体验、可用性研究和试运行的诸多反馈，产品经理就可以确定那些最重要的问题，并对特性的设计进行迭代，找出更好的解决方案。这时，确定优先级是产品经理最重要的职能之一。如果要求团队订正所有的 bug，构建所有的新特性，那产品永远也推出不了。产品经理需要斟酌所有的新需求，决定是把它们放到当前发布中，还是排到以后再做。

2.2.4　发布

开发过程完成后，产品经理要确保产品能平稳推出。各团队推出产品的过程也不尽相同，但通常都要做这样一些事情。

- 过一遍产品发布检查表。这里可能会需要关键利益相关者（比如法人代表）的最终确认，或者一个跟营销和运营团队相互协调的过程。
- 确保将要支持产品继续向前的团队准备好。对于在线产品而言，可能是客户服务团队；对于硬件产品而言，可能是制造团队；对于基于服务器的产品而言，可能是运营团队。
- 要为一切可能会出错的事情作好准备。随着发布将近，会不可避免地冒出一些紧急问题，产品经理要有如履薄冰的感觉。

在产品成功推出后，产品经理通常要向公司的其他人宣告产品的发布，跟团队一起庆功，然后准备好下一个任务再重来一次。根据团队的性质，产品经理可能会在产品推出后继续支持它，收集指标并根据用户的反馈进行迭代，也有可能会把产品交给另外一个团队去运营和维护。

2.2.5　产品类型如何影响产品经理的工作

产品经理真正的工作在很大程度上基于产品是什么。通过 DVD 或在线软件商店交付的软件和可以随时升级的在线软件有很大差别。此外，一个成熟产品的产品经理跟一个新产品的产品经理也可能会有很大差异。

1. 交付式软件

交付式软件是指像移动端 App 那样通过苹果软件商店交付或刻在 DVD 上交付的软件。交付式软件是独一无二的，因为在推出后很难升级。对于 Web 应用来说，你随时都可以发布一个新特性，如果出问题了，也可以迅速回滚。而对于交付式软件而言，第一次做对就非常重要。

因此交付式软件团队一般会有较长的时间线，并且产品需要更多的项目管理和团队间协作。功能规范也更重要，因为它的特性更正式，并且需要服务更多受众。用户研究和内部自我体验（使用软件的早期构建版本）也很重要，因为在产品对外发布之前你就得知道它是否好用。

擅于做项目管理并且有良好沟通技能的产品经理在做交付式软件时会有很棒的表现。对于想要维持工作与生活的平衡的人来说，交付式软件也很理想，因为这种软件通常不会需要你在几个小时内解决什么紧急问题。

2. 在线软件

在在线软件中保持旺盛的斗志非常重要。产品更新相当容易，所以一般发展得很迅速。团队一般不会等着把产品做到很完美，而是要经常推出些东西，看看它怎么样，然后再次推出。

大部分在线软件团队都会进行 A/B 测试（也称为多元化测试或在线实验）。在 A/B 测试中，新特性只会向一部分用户开放。实验组和控制组的行为会在后面进行比较，看新特性是否提升了用户体验。

因为做在线软件的公司会采集到更多的数据，所以产品经理一定要精通数据分析和设计实验。此外还要能承受较大的工作压力，因为服务器随时都可能宕掉，并且产品经理经常要快速作出决定。

3. 消费类产品

对于社交网络、照片分享应用和 Web 搜索这样的消费类产品，消费者就是普通大众，就像你自己、你的祖母和广大的工程师们。做消费类产品的好处是所有人都明白目标客户和用例。不过这同时也是一个劣势。

在面向消费者的产品上，工程师经常会有很多想法，不太依赖于产品经理拿出的特性和设计方案。产品经理经常扮演牧师和编辑的角色，指导产品向好的方向发展，而不是作出所有决定。

精通数据分析的产品经理可以很好地胜任消费类产品的工作，因为他们可以为自己的提案做出强健的案例，并且他们拿出的特性经常可以让公司所关心的核心指标得到改善。做消费类产品，你也很容易跟父母解释清楚自己究竟在做什么！

4. B2B 产品

对于在线广告或生产力软件这样的企业对企业（B2B）产品，客户是在另一个公司工作的人。对于这些产品，工程师知道他们不是目标受众，更倾向于依靠产品经理来了解客户。

B2B 产品的产品经理有时要考虑产品决策对收益会有什么样的影响。面对那些符合产品长期战略的功能和大客户当前就吵着要的功能，他们需要做出权衡。

喜欢做客户研究和营销分析的产品经理可能会喜欢做 B2B 产品。产品经理也往往能在这样的产品中发挥最大的影响力，所以他们会非常满意这样的工作。

5. 早期产品

对于比较新的产品，比如那些即将推出或刚刚推出的产品，团队经常会专注于交付最小可行产品（MVP）。因为这还不是应对全部挑战的时候，你甚至都不知道产品是不是符合市场和客户的需要。所以你要尽快地回答这些问题，并证明你的核心价值。

产品经理致力于砍掉不重要的功能，将产品剥离得只剩下最基本的部分。这样他们可以更快地推出产品，立刻开始一个了解客户真正需要（或者他们是否真的想要这个产品）的学习过程。有时这意味着推出的产品还没能如你所愿地进行打磨。

喜欢刺激，并且愿意以简便快捷的方式做事的产品经理能做好早期阶段的产品。对在早期阶段创业的产品经理而言，最让人兴奋的事情是把只有几个微不足道用户的产品发展成有庞大用户群的产品。

6. 成熟的产品

对于成熟的产品来说，比如市场上的领导品牌，大部分工作都是在已有产品上迭代，尽量完善它。产品经理通常会有前一版本的反馈，知道哪个区域最需要改进，可以把工作重点放在上面。

成熟产品的产品经理很重要的一点，是要确保自己不被微小的增量改进拖住。成熟产品的最大竞争对手经常是自己的上一版。与此同时，成熟产品通常都有充裕的时间在新想法上作出豪赌。

在成熟的产品上工作，最大的优势就是你已经有庞大的用户群了，你做出的每个改进都会产生巨大影响。从另一方面来看，很多公司都不愿意在成熟的产品上冒险，不敢作出大胆的改变。

想要在影响上百万人的产品上工作的产品经理，会喜欢成熟的产品。成熟的产品也能让你跟那些促成产品成功的人很好地学习。

2.3 产品经理最大的几个误区

因为产品经理还不是大家很了解的岗位，所以人们对产品经理是谁，以及他们是做什么的会有很多误解。这里是关于产品经理的十大误区。

1. 产品经理是项目经理

尽管有些产品经理承担了大部分的项目管理工作，但大多数人并不是这样的。项目经理关心的主要是时间节点和协调工作。尽管他们可能会负责收集项目需求，但他们在确定和选择需求上并没有多少话语权。

产品经理要负责找出问题和机会，挑选出会继续发展的那些问题和机会，然后确保团队能拿出优秀的解决方案，或者自己思考，或者跟设计师和工程师一起。所以对产品经理来说，对产品的感觉——能凭直觉判断出好产品和坏产品——才最为重要。

2. 产品经理属于营销部门

这个说法很有欺骗性。因为头衔没有固定的标准，有时一些营销岗位也被称为“产品经理”。但像谷歌、亚马逊、Twitter 和 Facebook 这样的公司，产品经理不属于营销部门；相反，他们通常归属于与工程相关的组织。

做营销的伙计们是要把用户吸引到产品中，而产品经理则负责判断用户在使用产品后会发生什么。

比如说，一位营销经理可能会发布消息并启动社交媒体活动，而产品经理则是给出新特性，并跟工程师一起推出它们。虽然营销人员会跟产品经理探讨这些帮助传递消息或打造品牌的特性，但他们不会去设想那些特性的细节，也不会跟工程师一起构建它们。

3. 你不可能大学一毕业就做产品经理

头衔中的“经理”这个词让很多人以为成为产品经理需要很丰富的经验。此外，因为产品经理要作许多对产品方向产生影响的决策，也让它看起来像是个高级岗位。

实际上，像谷歌、微软、Facebook 和雅虎这样的很多科技公司会直接校招产品经理。他们发现伟大的产品经理富有激情、智慧、强烈的客户关注和充沛的精力。如果你想成为一名产品经理，不要觉得你得先找份别的工作。

4. 产品经理就是写功能规范的

产品经理的工作跟工程师或设计师非常不同。人们对工程师的期望是让他们交付有效的代码，对设计师的期望是让他们交付草图和原型。而对于产品经理来说，只交付一份规范是不够的。

产品经理要负责照看整个项目，直到它成功完成。编写规范是一种沟通和推动项目的技术手段，但规范本身并没有内在的价值。很多产品经理并不借助规范来交流他们的想法，他们可以用谈话，还可以把想法画在白板上。也有一些产品经理失败了，他们虽然写了规范，但却没有参照执行来确保团队理解并实现其中的想法。

5. 产品经理只是召集开会的

有些人觉得产品经理的工作就是把关键的相关人员召集到一间屋子里作决定。好的产品经理并不只是被动地传递别人的看法。相反，产品经理要研究那个领域，并拿出自己的观点和框架来制定决策。

产品经理确实需要跟关键的相关人员碰面，了解他们的观点和优先级，但之后他们会把那些观点综合起来，取折中点，然后拿出一个令所有相关人员都能满意的推荐方案。在任何一次会议或谈话中，产品经理都需要呈现出所有不在场人员的利益。

产品经理可以减少队友们出席会议的次数，因为他们可以代表团队跟其他分组沟通，也可以找到其他有效的沟通方式。

6. 产品经理构建出的东西应该跟客户要求的完全一致

做客户研究并听取客户的要求很好，但那还不够。产品经理的视野要超越客户所讲的东西，洞见隐藏的需求和更深层的目标。

厨房用具公司 Oxo 询问客户他们的量杯有什么问题，客户说的都是杯子掉下去后打碎了，

或者它的把手太滑了。但他们仔细观察了人们如何使用量杯，看到人们先倒一些，然后弯下腰来看量度，再倒，再弯腰，再倒，再弯腰。

虽然没人要求在倒水的时候也能看刻度，但 Oxo 明白有这种需求。他们现在卖一种量度带有一定角度的量杯，这样你在往外倒东西时也能看到刻度线了。

7. 产品经理确定日期

就像谷歌的产品经理 Nundu 说的那样："日期不是产品经理定的，是工程师定的。"作为产品经理，你可以告诉团队你想让他们构建什么，然后他们会告诉你需要多长时间做出来。如果时间太长，你可以要求他们代码写得快点，但那没用。

相反，如果你要满足外部的截止时间，就需要进行取舍或协商。或者砍掉特性，或者找到并行工作的办法，并找更多的人来帮忙。有时你甚至可以更聪明，能找到办法减少工程师的工作量，比如让他们省去没必要的会议，或者让他们暂时少花些时间去参加新员工面试。

如果你不相信工程师的估算，给其他团队承诺的时间比工程师认可的时间要短，那会立刻破坏你跟团队的关系。

8. 产品经理是老板

有些试图推销产品经理岗位的人会说你就是团队的 CEO。实际上，产品经理对团队没有直接的权威。团队绝对没义务按照产品经理的指挥做事。

相反，产品经理要在自己没有权力的情况下产生影响，在团队中积累威信，清晰地沟通，收集数据搞研究，靠自己的说服力领导团队。只有当团队确信大家的目标一致，并且产品经理能够帮助他们更好地达成目标时，他们才会跟随产品经理。

作为产品经理，一定不要试图告诉每个人他该做什么，不要干涉设计师或工程师的工作。设计师有权掌控产品的设计。产品经理要明白那些选择，以及它们在总体体验上的影响。如果没能达成一致，产品经理可以提出来，但不应错误地试图去控制那些决定。

9. 想法比执行更重要

一些刚刚从事产品经理工作的人有时会觉得想出好主意是工作中最重要的。实际上，想法的执行要重要得多。团队中的很多人都能拿出很多好想法，但通常细节才是困难之处。

作为产品经理，必须广泛征集想法，并把它们变成切实可行的。产品经理要考虑各种极端情况，确定出把一个想法变成现实所有的小步骤。这经常需要你亲自出手：在服务器上运行你们的代码，说服其他团队优先处理你们所依赖的工作，持续不断地使用产品，找出并打磨好所有粗糙的地方。

10. 产品经理可以说“那不是我的工作”

团队中的大多数岗位定义得都很干脆，而产品经理的角色流动性比较强。如果你是产品经理，那你的工作就是其他人覆盖不到的所有事情。

作为产品经理，你要为产品的成败负责，没有什么工作是你不应该做的。如果有一项工作没人想做，你就需要想办法解决它——即便那意味着要由你来做。如果你放任不管，就不会有其他人来解决它。

2.4 项目经理和专案经理

> **注意**：微软那个被称为专案经理（Program Manager）的岗位跟大多数其他软件公司的专案经理不同，而是比较像之前描述的产品经理一职。

有很多岗位都跟产品管理相关，并且这些岗位之间的界限可能很模糊。如果你希望应聘项目经理或专案经理的职位，看看能否找到在那个公司做过这个工作的人聊一聊，弄懂这些要求之间细微的差别。

项目经理要确保他们的项目能按照预算及时交付，不管客户是谁，都要让他们满意。专案经理（不包括微软的专案经理）也差不多，不过他们通常掌管一个长期运行的专案，而不是一系列设定了截至时间的项目。

软件公司经常会有项目经理领导面向内部的团队，比如基础设施项目和运营专案，或领导重点面向单一客户的咨询团队。这跟产品经理相反，产品经理通常是带领团队构建面向客户的产品。然而这不是一个硬性规则。

工作

关于这两个岗位，有两件事是很独特的。首先，项目经理有非常清晰明确的客户或目标。

基础设施和运营团队的项目经理，把公司内的其他员工当成自己的客户。其他员工通常都非常清楚自己想从项目中得到什么。

咨询团队的项目经理为单个客户服务，这个客户已经签署了一个包含每个项目细节的工作说明。

在所有这些情况下，研究、规划和设计阶段都相当简单直接，或者已经有人做好了。对项目经理来说，产品设计通常不是他们的主要工作，所以你可能会（也可能不会）被问到有关于产品

设计的问题。

还有，因为客户非常关注项目结果，所以沟通和期望值设定非常重要。项目经理要乐于汇报团队的进展情况。项目经理一定要面向细节，跟团队打成一片，这样你才能回答客户提出的问题，并且能在出状况时做出解释。

第二个独特的部分是项目管理和专案管理很大一部分工作是预算和资源管理。项目经理经常要在成本中心工作，所以他们的精力主要放在运营效率和降低成本上，同时还要保证质量，而产品经理通常会屏蔽这些方面。

下面是项目经理和专案经理要做的预算和资源管理方面的一些工作：

- 澄清目标并收集满意度指标；
- 确定完成项目所需的人员和技能；
- 设置项目管理工具、规划和过程；
- 发起状态会议，收集状态报告；
- 分析数据找出机会；
- 确定并实现修改以提升效率；
- 管理客户做出的修改；
- 想办法让项目保持正轨，即便有事情出错时也不例外。

每个公司及其相关角色对项目经理的要求都不一样。有些团队要求有项目经理认证资格，而有些没有要求。公司一般都爱找之前有过项目管理经验以及拥有良好沟通技巧的人。

第 3 章
公　　司

产品经理候选人最想问的问题之一是：“这些公司有什么区别？”尽管产品经理在很多公司中的职责都非常相似，但实际上他们的日常工作可能会有很大差别。

3.1　产品经理的职责有何不同

我们跟亚马逊、苹果、Facebook、谷歌、微软、雅虎以及很多创业公司的产品经理都聊过，以便了解每家公司产品经理的职责是什么样的，每个公司有何独特之处。

我们了解到，各家公司的差异之处大体表现为：部门之间的透明度、工作与生活的平衡度，以及对于有技术背景的候选人和有商学院背景的候选人，各公司的评价侧重。产品经理的职责在工作范围上也不一样，具体表现在对产品定义、设计、产品策略和项目管理等工作的参与度上。

3.1.1　透明度

有些公司非常透明，比如谷歌、Facebook 和雅虎，很多地方都可以直观地看到其他团队在做什么。而有些公司，比如苹果和亚马逊，隔离程度比较高，每个团队只专注于他们自己的工作。

在比较透明的公司，经常可以见到产品经理在不同的团队间穿插，而且团队间的协作也是产品经理的一大块工作。在比较隔离的公司，岗位变动的情况比较少见，并且通常强制的多团队任务也比较少。

3.1.2　产品经理与工程师的比例

产品经理与工程师的比例也可能会有很大差异。微软有很多产品经理，在某些团队中的比例高达 1∶3。其他公司比较常见的是 1∶10 的比例。谷歌和 Twitter 的产品经理与工程师比例非常低。产品经理在日常工作中与工程师的合作有多紧密，以及能负责多大的产品，在很大程度上取

决于这个比例。

在有很多产品经理的公司，会有很多协作工作，并且有很多机会可以跟经验更丰富的人学习。在产品经理少的公司，很有可能可以负责一个很大的领域，并且可以独当一面。

3.1.3 产品策略

在不同的公司，产品经理在定义产品策略时的职责也不同。有些公司的策略是“自底向上”的，重要决定经常来自开发人员和产品经理。有些公司的策略是“自顶向下”的，他们的方向通常是由高管和产品经理定的，开发人员只是负责实现。

在谷歌、Facebook、雅虎和亚马逊，产品经理会深入参与产品策略的制定，决定将产品带向哪个方向，以及何时开始新的行动。人们希望产品经理可以考虑他们团队的策略，比如专注于哪些客户和领域。他们会从很多人那里吸取建议和看法，但最终他们要呈现一个完整的计划。这些公司的产品经理一般都可以轻松得到一个工程师团队来实现他们的想法，最起码也能做到实验阶段，在产品推出之前完全不用高管批准。

而微软和苹果的策略一般是自顶向下的，不同的产品经理都按照这种策略执行。这并不是说产品经理在自顶向下的公司里影响不了策略的制定，但在级别升高之前，很多产品经理确实没有这方面的影响力。在这些公司，只有那些非常出色的产品经理才能把他们的想法推销给高管，赢得上层支持，从而影响产品策略。

3.1.4 企业文化

另一个巨大差别是企业文化。在谷歌、微软、雅虎和 Facebook 这样的公司里，上班时的心态是无忧无虑的。这些公司有引以为傲的福利，比如免费的食品和饮料，甚至免费的按摩服务。并且因为人们可能会工作很长时间，所以他们总是说工作质量要比工作时长更重要。

另一方面，在像苹果和亚马逊这样一些公司的文化中，令员工引以为傲的是他们工作的努力程度。在这些公司里，员工应该会工作很长时间，并且崇尚节俭。这些公司的员工受到公司使命的激励，愿意在周末加班，或者在深夜接听电话。做一款优秀的产品可不是件轻松的事。

3.1.5 他们招什么样的人

亚马逊喜欢招 MBA 担任产品经理一职，并且不觉得技术背景是必需条件。和少数几家公司一样，他们不会招大学应届毕业生做产品经理。然而，他们确实愿意接受应届毕业生担任专案经理（Program Manager）或技术专案经理（Technical Program Manager），这个岗位相对来说更偏重于项目管理，而不是产品设计。

苹果既招工程专案经理（EPM，Engineering Program Manager），也招软件产品经理，所以对于有电子工程或计算机科学背景的人来说，它是个不错的选择。苹果招聘应届毕业生做工程专案经理，在这个岗位上一般不会招 MBA。

Facebook 是技术性最强的公司，要求所有产品经理都有技术背景。这家公司崇尚它的创业时的“黑客”文化，并且有相当数量的产品经理是它所收购的公司的创始人，还有很多之前是谷歌的产品经理。他们招聘应届毕业生做轮岗产品经理，该职位包含 3 次为期四个月的跨团队轮岗。

谷歌喜欢招应届毕业生，通常是主修计算机科学专业的，开始要参加一个助理产品经理（APM，Associate Product Manager）计划，这是一个为期两年的轮岗计划。谷歌的一些产品经理有 MBA，但更重视硕士或博士学位。

雅虎主要是招有经验的产品经理，但现在也有一个助理产品经理计划来招募应届毕业生。雅虎看重有技术背景，并且能很好地跟工程师沟通的人。他们在寻找那种有良好产品感觉，充满激情的人，这种人既要有创新精神，又要能务实地交付有亿万用户的软件。

微软招的专案经理既有应届毕业生，也有经验丰富的人，并且更喜欢有技术背景的，尽管不一定必须是计算机科学专业的。另外，微软会在产品经理岗位上招聘 MBA，产品经理在微软是个营销岗位。微软的国际化招聘做得很好，在美国境外招了很多产品经理。

3.2 谷歌

谷歌的结构反映了它的创始根源。谷歌对创新充满热情，并且十分推崇将伟大的想法变成现实的文化。谷歌的愿景是自底而上的，并且团队经常是由工程来驱动的。产品经理专注于策略、分析和促进工程师团队。

谷歌的独特之处在于各组织之间有着令人难以置信的透明度。任何一个全职的谷歌人都可以看到大部分代码和文档，并且高管团队会处理所有谷歌人在每周的全员“TGIF”例会上提出的问题。谷歌的产品经理在工作时经常会切换产品团队。

3.2.1 他们招什么样的人

谷歌要的是那种喜爱技术，能够自我激励，并且有创新精神的人。有 MBA 或工作经验超过 4 年的人都可以应聘谷歌的产品经理，而那些经验不足 4 年的人则可以申请助理产品经理。

助理产品经理计划是为应届毕业生准备的为期两年的产品管理精英培训计划。助理产品经理会被分配到全公司各团队的重要产品管理岗位上，不仅有各种培训与交流的机会，一年后还可以轮岗。另外，谷歌还会为他们提供国际商务旅行，拜访遍布世界各地的谷歌人和客户。

3.2.2　他们做什么

谷歌有很多产品，产品管理岗位在各团队中可能会有很大的变化。尽管在所有这些岗位中，产品设计思路和分析技能都是产品经理非常重要的必备技能，但对于不同的产品，如谷歌搜索、谷歌关键字广告（AdWords）、Gmail、Android、YouTube、Google Plus 和谷歌地图等，其侧重点也会有很大差异。

搜索团队非常依赖搜索技术，工程师会主动去开发新的算法。对面向广告商的团队来说，产品经理则负责收集客户需求，并向团队其他人传达那些需求。在像谷歌 Plus 这样的团队里，设计师是团队的灵魂，而面向开发人员的团队可能根本就没有设计师。

产品经理在谷歌是独立工作的。产品经理加入团队，然后由他和整个团队决定要构建什么。产品经理在谷歌的第一个项目通常都是探索未来的工作方向。

很多产品只有一个产品经理，那些不止一个产品经理的产品，工作一般也划分得很清晰，每个产品经理都有自己完整的领域。在日常工作中，谷歌产品经理会跟工程师团队以及设计师紧密合作。很多新想法都是产品经理、工程师和设计师在白板上写写画画时讨论出来的，然后就能快速搭建出原型。

谷歌极其看重产品经理的分析技能，因为数据分析是产品经理的一大块工作。产品经理会频繁地检查使用日志，以便为搜索和广告部门拿出新的项目想法。团队做出东西后，谷歌很容易就可以把它拿给一小部分用户试用。然后数据就会源源不断地进来，产品经理要分析这些数据，或者跟数据分析师合作，看看所做的修改能否称得上是一项改进。

谷歌的产品经理有一大块工作是让项目初具雏形，能够推出。因为哪怕是很小的改变，也会被千百万用户看到，所以一定要让所有东西恰到好处，包括 UI 和算法。项目必须满足安全、法务和架构要求。在得到最终核准之前，很多项目都要经过反复的研讨与改进。

3.2.3　创新

为了鼓励创新，谷歌有个“20%时间”计划。根据这一政策，工程师和产品经理可以把他们20%的时间用在跟公司相关的副业项目上。启动 20%项目不需要任何人的批准，只管开始做就行。你可以把你的项目提交到内部网站上，招募其他人加入你的项目。Gmail、谷歌新闻和 Orkut 之类的很多大型产品就是从某些人的 20%时间中诞生出来的。

所有这些创新和自由度都让谷歌成为了产品经理心目中的梦想。如果你有一个自己特别喜欢的项目，你不仅有时间和自由自己做，并且身边还有很多优秀的工程师，可以让他们在富余的时间里帮你把它做出来！

3.3 微软

微软的产品经理岗位始于 20 世纪 80 年代，当时他们意识到在营销和工程师团队之间需要增设一个岗位，安排专人对照客户的需求实现产品的可用性。

微软的专案经理（Program Manager，相当于产品经理）一职在领域和影响力上都是非常独特的。他既要充当业务分析员、项目经理，还要提供创意。微软还是产品经理对开发人员比例最高的公司之一。综合所有这些因素，使得微软的产品经理成了一个实践性非常强的岗位。团队经常是由产品经理驱动的，而所有面向用户的决定都由专案经理来做。

作为一家比较老牌的科技公司，微软非常注重员工的职业发展规划。他们知道员工想要明确的职业进程以及完善的成长机制，因而建立起一种多元化的成长机制，让员工的发展方向不仅仅局限于成为团队领导，还有可能为更大的产品负责，为更多的产品策略负责。很多员工都希望能在微软走完自己职业生涯的全程。

3.3.1 他们招什么样的人

微软想招有大局观，能解决问题，能把事情做成的人做专案经理。

微软的 Uche 说："我们想要那种喜欢刨根问底，能从多个角度看问题的人。作为产品经理，你将会扮演很多角色，所以你的思考方式比任何特定的技术技能都重要。"

微软有两个跟专案管理有关的岗位一般都是由 MBA 担任：产品经理和产品规划师。微软的产品经理归属于营销团队，要发现市场机会，并为推动这些机会开发市场策略，且都是专注于当前的发布版本。产品规划师的目光则要放得远一点，要发现市场和技术趋势，拿出新的产品场景。

3.3.2 他们做什么

微软的产品愿景和策略通常来自上层，然后自上而下贯彻执行。

比如说，主管微软 Office 的副总裁（VP）会跟专案经理小组一起工作，创建和分享愿景文档，其中包含下一版要关注的主要领域。每个 Office 产品的负责人都要定义自己负责的分类产品的愿景，跟 Office 的总体愿景保持一致。然后每个团队的负责人都在 Office 的总体愿景及分类产品愿景的基础之上，再开发各自团队负责的功能领域的相关愿景。

这样做的结果是微软的产品策略在一个产品部门内部非常有聚合性。所有团队都会觉得他们在朝着相同的目标一起努力，并且几乎不太可能有两个团队同时在做相互竞争的特性。随着职业生涯的发展，你负责的策略也会越来越大。

由于采用这种自顶而下的愿景，所以很难在产品推出周期中间发生方向上的巨变。即便想法

很好，也很难找到有空闲时间的开发人员来构建它。换句话说，你在做的通常是每个人都认可的重要特性。这意味着你只能把精力用在构建优秀的产品上面，而没法去说服管理层推出你做好的伟大产品。

微软会让新产品经理负责一个特性领域，跟进团队的愿景，并且会给他很多的自由度和职责，以期让产品的那些部分变得更棒。一旦产品经理在职业生涯早期证明了自己在团队中的价值，就会承担起越来越大的责任。

微软的核心特性团队包括一名开发人员、一名专案经理和一名测试人员。最近有些团队又给核心特性团队加了一名设计师。核心特性团队在产品经理的领导下，共同决定构建的内容。在很多团队里，产品经理都先要编写只有一页的“规范”，其内容是对目标和用例高度抽象的描述。

在跟产品的其他专案经理一起探讨过这个只有一页的规范后，产品经理可能会把这个简要的规范扩展得更为详细，准确描述该特性的工作方式，从抽象的流程到具体的错误消息文本。产品经理可能会跟设计师合作，也可能会自己把交互设计都做了。这取决于团队的具体情况。

规范评审完成并开始实现后，自我体验（内部试用软件的早期版本）就变得非常重要。特别是交付周期比较长的团队，微软的产品经理会从公司的其他人那里收集反馈。反馈进来后，产品经理要给 bug 和新特性的想法排优先级。

在负责核心特性的工作之外，产品经理还会承担一些团队内或是团队间的职责，比如负责产品的项目管理日程，或是在产品推出前进行检查分类，以决定究竟应该修订哪些 bug 或实现哪些特性。

随着微软把自己调整成设备和服务公司，很多团队已经用更敏捷和迭代式的工作方式取代了严重依赖文档的工作方式。团队推出产品的速度更快，也更频繁了。A/B 测试正变得更加重要。

3.4 苹果

苹果的组织结构是自顶向下的，各部门独立性较强。产品方向或者说愿景被牢牢地抓在高管团队和设计师手里，而公司的其他部分就像运转良好的机器那样只需执行这个愿景即可。在苹果，工程项目经理和工程专案经理（都被称为 EPM）是产品的领导，维持着整个机制。

3.4.1 他们招什么样的人

苹果找的是那些跟苹果的产品形影不离的人。尽管很多公司为自己能够提供健康平衡的工作/生活方式而自豪，但苹果希望它的员工对产品充满激情，将其视作他们的生命。

对于 EPM 一职，他们找的人通常会有很强的科学及数学背景（那样他才能“理出头绪”），

还要能有良好的风度（那样他才能有适度的自信）。此外，他们还希望此人的技术能力达到能够顺畅地参与讨论，但没必要从事任何具体的技术工作。

因为苹果有很多硬件项目，所以除了软件 EPM，他们还要招聘硬件 EPM 和系统 EPM。软件 EPM 通常会有计算机科学背景，而硬件 EPM 可能会有其他工程领域的背景，比如电子工程或机械工程。

在苹果，什么样经验水平的 EPM 都有，既有应届毕业生，也有具备 15 年行业经验的业内老兵。苹果的大多数 EPM 都来自工程或技术岗位，或是在其他公司担任过 EPM，很少有经管专业的人或 MBA。

苹果还有产品营销经理（Product Marketing Manager）一职，有时候称作产品经理。对于产品营销经理，他们要找的是有商业或营销背景的人。这个岗位经常会聘请 MBA 毕业生。

3.4.2 他们做什么

在苹果，创意最初的萌芽可能是自顶而下的，也可能是自底而上。在 EPM 的推动下，经过高管团队或高级管理层一系列的评审与研讨后，最初创意逐渐成形。产品经理要进行客户研究，查看市场趋势，以确定下一版的策略。产品一旦得到批准，EPM 带领工程师团队构建产品，创建开发日程，促成多个职能团队间的沟通，并带头解决问题。

在苹果做产品，需要协调很多团队，而 EPM 就是沟通的枢纽，要确保事情按日程运转，如果没有的话就要负责解决问题。比如说，在做硬件设备时，要协调机械设计团队、电子设计团队、外协厂商，以及运维团队。

苹果总部库比蒂诺的 EPM 一天要做很多沟通方面的工作：报告团队的进展，了解其他团队的状态，跟高管一起回顾并讨论当前的状态。通过这些讨论，EPM 负责找出潜在问题及其解决办法。

对于软件 EPM 而言，日常工作的一项重点内容是测试每日构建，寻找妨碍后续工作的问题，并确保它们能及时解决。

系统 EPM 的重点是交付整个产品。他们会制定规划，带领数十名苹果工程师到海外做原型构建。除了在外协厂商那里代表苹果工程团队之外，系统 EPM 还要与制造商一起进行故障分析，直到将故障消除，协调开发对硬件的不同需求，消除妨碍原型构建的故障。系统 EPM 是苹果内部项目的领导者，负责团结其他 EPM 和工程师来交付产品。

硬件 EPM 的工作重点是产品的硬件交付，包括电路板和布线方式。他们会跟电子工程团队、产品设计团队、芯片团队以及外部供应商紧密合作。他们日复一日参与着芯片计划和供应链管理的工作中，还经常就困难的跨职能设计决策方面的问题，使各方达成共识。

3.5 Facebook

Facebook 是一家斗志昂扬，以技术为中心的公司。他们几乎没几个产品经理。很多团队一开始都没有产品经理，只有在后来明显需要时才引进一位。甚至在团队的产品经理不止一位时，团队的领域之大，也足以让每个人独当一面。

3.5.1 他们招什么样的人

Facebook 有一群特立独行的产品经理。

Facebook 寻找技术高超又有创新精神的产品经理。在 Facebook，所有的产品经理都要求会编码（或者最起码学过基础知识），要通过 Facebook 新手营的考核，这是一个为期 6 周的培训计划，产品经理和工程师要在其间学习如何使用工具和解决 bug。这符合 Facebook 自己动手的企业文化，产品经理经常会自己动手写出最初的产品原型。

当 Facebook 为了招募员工而收购一家公司时，人们称其为收购式招聘。Facebook 的收购式招聘通常寻找的是小团队，一般不超过 10 个人，大部分都是技术人员。公司的创始人或 CEO 经常会被带过来做产品经理。

应届毕业生和没有产品管理经验的人会加入轮岗产品经理计划。这个计划为期一年，包含 3 次在不同团队的轮岗，每次为期 4 个月。

3.5.2 他们做什么

Facebook 会有一些已经制定好的计划，但很多创意还是来源于对上网用户的观察——人们如何使用网站，以及他们会遇到什么样的问题。产品经理会注意到需要关注的领域，并把故事板和提案放到一起来阐述他们想做什么，包括预期结果。因为产品经理如此自由，所以你必须知道如何把自己的想法置于公司的整体框架中，并展示出它跟全局的匹配度。

然后团队会做出一个原型，先在一个小范围市场上测试，看能否达成预期结果。如果一切顺利，正式的项目开发就会进入议程。产品经理会同马克·扎克伯格（被大家称为“扎克”），或者部门负责人一起，评审提案以便得到批准。然后团队会引入设计师，在内部启用这个特性，开始从 Facebook 的其他同事那里收集反馈。

在开发过程中，产品经理会通过产品评审进行迭代。这里一般没有太多项目管理上的开销，但只要工程师一签入代码，观察日志的产品经理马上就能看到新代码。一切就绪后，产品经理会协调一个部署计划，会同营销部门一起推出产品。

3.6 亚马逊

亚马逊的文化是由他们的 14 条领导力原则指引的（参见附录中的“亚马逊领导力准则”）。尽管很多公司都有自己的原则体系，但亚马逊领导力准则在招聘和日常工作中起到了非常重要的作用。人们会在会议中引用领导力原则来协助作出决策，会在面试中寻找具备这些品质的候选人。

3.6.1 他们招什么样的人

亚马逊有很多业务，既有正在启动的新举措，正在进行的技术性业务专案，也有非技术性业务专案。他们有不同的团队，分别专注于消费者、出版商和开发人员等不同受众。基于以上情况，亚马逊有很多跟产品和专案管理相关的职位。

产品经理是产品的主人，他们专注于产品的愿景。亚马逊喜欢招 MBA 做产品经理，喜欢招刚从商学院毕业的人。亚马逊跟其他很多公司不同，他们不要求产品经理有技术背景。

技术专案经理（TPM）负责技术项目的日常执行，他通常需要有很强的技术背景，可以是直接从学校毕业的，也可以是从工程师职位上转岗来的。TPM 跟工程师的合作非常紧密。

专案经理则负责非技术性项目的项目管理，比如运营上的项目。专案经理可以是具有任何背景的人，但会 SQL 是加分项。亚马逊寻找的专案经理要聪明、敏捷，并且耐压能力强。专案经理负责管理和改善团队的工作流程。

3.6.2 他们做什么

产品经理决定团队的愿景和路线。亚马逊极为看重“客户至尚”的理念，很多新创意都是在确定客户需求的过程中出现的：这其中既有直接跟客户交谈而得，也有通过对用 SQL 查出来的数据进行分析而得到的。亚马逊是由数据驱动的，所以需要产品经理有很强的分析技能。

产品经理想出一个创意后，他会把它和一个业务案例一起放在备忘录里，也被称为“故事”。这个文档中会包含支撑它的推荐及分析的详细信息，特别是关于影响和理由的数值。亚马逊用文档而不是演示文稿来介绍新提案，因为文档会迫使作者更精确，并且能显现出清晰的思路。

经过最初的几轮修改和批注后，提案会在会议中分享给上面的管理层。会议一开始，所有人都要安静地看完文档。这个过程可能一开始看起来会有点儿奇怪，但它可以确保每个人都有时间阅读提案的细节，并且能得到所有人的重视。大家都看完后，就可以开始进行提问。

提案可能还会再过几轮，关键是要能对从输入到预期结果进行准确的规划。提案通过之后，团队就可以开始构建它了。很多团队采用敏捷开发，产品经理把控产品走向，负责编写用户故事和团队的待办事项。

等产品准备好可以推出后，产品经理会跟营销团队一起合作，并准备把它交给运维团队，由他们在一个持续的基础上运营这个专案。

3.7 雅虎

雅虎有很强的传统：曾几何时，“雅虎”对很多人来说就是“互联网”的代名词。在过去几年，这家公司经历了一系列的转变，但在 2012 年，玛丽莎・梅耶（Marissa Mayer）被任命为公司的 CEO，从而为公司注入了新的活力和激情。Web 流量上来了，很多最近发布和上线的产品都受到了好评，并且内部和外部认知上都有明显的改观。

这样明显的改变让雅虎的产品经理都感到兴奋。他们现在可以专注于用户的所思所想，把他们的需要放在第一位。由梅耶这位此前的谷歌产品经理掌舵，雅虎开始密切关注产品研发，希望能做出用户喜爱的产品。

3.7.1 他们招什么样的人

尽管没有要求，但雅虎的产品经理一般都有计算机科学的学位。雅虎寻找有很强技术背景的天才，或者看在学校时的成绩，或者有在顶级软件公司任职的经历。

现在雅虎有各种各样的产品经理：有些已经在雅虎工作很长时间了，有些则是刚从其他科技公司跳槽过来的，还有些是应届大学毕业生。尽管雅虎的每位产品经理都有不同的背景和经验，但他们都有一个共同的目标，那就是让雅虎的产品及其用户体验更上一层楼。

为了招募有天分的应届毕业生，雅虎启动了助理产品经理（APM）计划。这是一个为期两年的轮岗计划，其间由来自公司各部门的人定期进行培训与讲演，中间还有一次全球的商务旅行。雅虎的 APM 计划有两个独特之处：在 APM 开始工作之前，他们就了解自己所在的团队，并且组织者致力于将所有的 APM 推向更大、更有挑战性的岗位。

3.7.2 他们做什么

雅虎的团队经常以三人成组，通常由一名工程师、一名产品经理和一名设计师组成，由其中的一个人（一般是产品经理）担任团队领导。

产品经理负责设定产品的总体方向和策略，确保用户体验，并且形成能够长期发展的良好态势。这包括用户体验和长期盈利计划，确保团队不再承受寻求短期回报的压力。

雅虎有很强的协作文化，公司里随处都会有创意冒出来。为雅虎的核心产品做出合适的移动端体验是公司工作的关键驱动力。

雅虎的大多数产品经理技术都非常强，在公司内部也很受尊敬。产品经理全方位地参与到产品的整个生命周期中，他们做实验，跟多个职能团队紧密合作，并且监督整个推出过程。产品推出会有官方的评审过程，在把产品开放给客户之前，帮着把好产品的质量关。

雅虎的团队动作快得多，并且很多采用的都是敏捷方法。很多团队的迭代都比较短。产品经理现在抛开了冗长、详尽的规范，可能只用一句话就把用户故事写好了，然后就由工程师作决定，把它做出来。

3.8 Twitter

Twitter 是一家发展非常快的公司，致力于发展未来的沟通方式。因为 Twitter 在用户和员工人数上增长得特别快，所以这家公司一直在改变和进化。这种增长还为很多四处寻找提升职业空间的人打开了一扇大门。

3.8.1 他们招什么样的人

Twitter 喜欢雇用那些对 Twitter 有激情，喜爱他们的产品、公司或使命的人。

Twitter 有两个相关的岗位：产品经理和技术专案经理（TPM）。产品经理通常在面向客户的团队工作。技术专案经理通常做一些跟平台或基础设施举措相关的工作，经常要跨越多个团队。产品经理岗位的工作更偏重于产品设计，而技术专案经理会参与更多的项目管理。

对于技术专案经理一职，Twitter 寻找那种有技术背景且有软件交付经验的人。此外，他们还希望候选人有“把事情做好”的心态和良好的沟通技能。很多技术专案经理都有专案管理经验，但也有很多人具有咨询、工程管理和技术架构的背景。

对于产品经理一职，Twitter 寻找那种能够专注于用户需求，但又足够灵活，能够应对快节奏的人。因为 Twitter 的产品经理与工程师的比例相对较低，他们一般会雇用能跟上工作进展，经验更丰富的候选人。

3.8.2 他们做什么

Twitter 的产品经理有着多重的职责。团队是由产品、工程和设计人员组成的。产品经理要参与概念和路线图的制定，还要寻找 bug，给特性排优先级，不仅要执行当前项目，还要考虑将来。产品经理要充当特性领域和公司其他领域之间的接口。

技术专案经理一职在 Twitter 相对来说比较新，并且从某种程度上来说还没完全定型。他们有几种不同的技术专案经理，Twitter 正在为他们打造各自的职业路线。

有些技术专案经理负责发布管理：看管 Web 和移动版的发布过程，确保构建被合并进来了，管理自我体验过程（Twitter 的员工在这个过程中试用新的构建），确保 bug 得以解决。

有些技术专案经理管理的专案范围更为广泛，跨越多个团队，总数可能有几百人，并且会包括来自其他公司的外部团队。还有一些技术专案经理被认为是在跨团队主题领域（比如说敏捷过程）内的领导者，而另外一些技术专案经理则会被安排到单个平台或基础设施的团队里。

在 Twitter，创意的来源无处不在，管理层、产品经理、工程师、设计师以及客服，人人都可以提供创意。Twitter 用每季度一次的黑客周为此提供支持，在此期间，只要跟 Twitter 有关，人们想做什么都行。有很多伟大的创意都来自黑客周，人们可以利用这段时间按照各自的想法做些好玩的东西。

Twitter 每个季度都会做计划，团队要为他们自己的目标（因为 Twitter 喜欢用鸟取名，所以这个被称为“海鸥”）负责。在实现过程中，团队要一直做自我体验（先在内部尝试新特性）并在每周的例会上呈现他们做的东西。

Twitter 会做很多的 A/B 测试，并且都是数据驱动的。当然，过于注重数据也不太可取。团队以成功指标为参照，利用指标比对来了解事务的运作状况，然后进行必要改变，但他们仍要确保快速行动，并保持住产品的灵魂。

3.9 创业公司

对产品经理来说，创业公司可能非常棒，因为他们可以施加自己的影响力，并能在参与的过程中亲身塑造自己的角色。跟在大公司相比，产品经理在创业公司确实要扮演很多角色，并且要找到积极有效的解决办法。

要判断一家创业公司的企业文化，最好的办法就是看创始人、产品经理，以及早期员工的职业经历。因为产品经理没有一种成熟的定义，团队一般会根据以前就职的公司的岗位状况来定义产品经理。前苹果员工创办的公司一般会具有苹果文化的某些特征，而前谷歌员工创办的公司也通常更像谷歌。

一旦你准备深入了解一家创业公司，请产品团队的人喝杯咖啡也是很有帮助的。在创业公司，所有员工都要经常参与招聘，所以他们一般也都爱和别人聊聊。另外，因为他们在创业公司工作（非常辛苦），所以他们可能也需要喝杯咖啡！

3.9.1 他们招什么样的人

创业公司要找的人在资历和技能上差异很大。他们一般更喜欢找那种有软件交付经验的人，除非公司已经有一个大型产品经理团队，否则一般不愿意考虑应届毕业生。

对创业公司来说，文化契合与激情对公司来说十分重要。当你想加入公司时说“我想跟伟大的团队一起工作，有巨大影响力”时，大公司可能会对这个理由表示满意，但对创业公司来说却行不通。他们想知道你为何喜欢他们的领域，而且陈述要力求具体。

找工作

如果你想在创业公司找一份早期产品经理的工作，最好能描绘一下你对该行业未来愿景的预期，给他们留下一个强烈的印象。Indix 的现任营销主管 Shalendra Chhabra（Shalen）曾用这种办法在两家不同的创业公司成功地应聘为早期产品经理，尽管他此前的工作经历跟这两家公司的业务并没有太直接的联系。

他找工作时，跟一个朋友讨论了各家创业公司的情况，归纳出一份选择清单。Swype 让他深感兴趣，这是一家为触摸设备做屏幕软键盘的公司。他写了封自荐信，说明了一下他对该领域未来前景的展望：触摸设备会变得极其流行，他们应该会将自己的产品应用于不同的语言情境中，并且会增加预测式输入这样的新特性。

在这份自荐信中，他描述了会怎样利用自身的背景和技能为公司带来价值。然后，他被朋友的朋友介绍给了公司的某个人，送出了他的自荐信。

写自荐信既花时间又费工夫，但结果证明这样做是完全值得的。自荐信帮他吸引了公司的注意，他以自己的方式给公司留下了第一印象。在他去面试时，因为面试官已经传阅过这份自荐信，所以他已经积累了一定的信誉。Swype 最终有一个非常成功的结局，被 Nuance 以 1.025 亿美元收购。

推荐对所有公司来说都很重要，但对创业公司来说更是如此，因为他们更不愿意冒险。要知道，在只有几个产品经理的公司中，任何一个都会对组织产生巨大影响。在公司找一个你认识的人会很有帮助——即使你跟他们不是特别熟。

除了员工推荐的，创业公司还会找他们的投资人、顾问和董事会成员推荐的人。你还可以考虑通过风险投资公司或者 AngelList 这种网站的渠道来接触他们所有的投资组合公司。

最后，很多创业公司都会从公司内部起用第一位产品经理。

比如说，Paul 是 TechExcel 的技术支持，他凭借自己跟客户的联系，提出了建设性的产品建议。然后 CEO 就让他把握产品特性了。

还有一个例子，Jesse 原来是 Venmo 的工程师，他对可用性测试颇有兴趣。在重新设计期间，他表现出了在团队组织方面的领导才能，然后公司创始人就让他做了第一个产品经理。

3.9.2 他们做什么

在创业公司做产品经理跟在大公司做产品经理差不多，仍然是要跟工程师和设计师合作，要对产品的成功负责。创业公司的产品经理帮助定义路线图，规范特性，检查 bug，分析实验，并全程负责产品的发布工作。

创业公司的产品经理跟大公司产品经理最大的不同之处在于规模。因为创业公司没有庞大的管理结构，产品经理自然成了公司的重要领导。另外，创业公司的资源也比较少，所以有更多的“空白”等着产品经理填补，更多时候会需要积极有效的规划与执行。

创业公司的产品经理通常会对公司的企业文化、工作流程和路线图产生很大影响。Venmo 的愿景宣言甚至都是由它的第一位产品经理 Jesse Bentert 促成的。

> 第一次做产品经理时，我意识到为了让大家齐心协力，团队需要一个使命。有一天我把大家组织到一起，集思广益，用一句话写下了 Venmo 的愿景：“通过支付连接世界并赋予力量。”

对于创业公司的产品经理来说，定义岗位职责是一项很重要的工作。因为你加入的不是经验丰富的产品经理团队，所以得到的指导与帮助也比较少。此外，大公司里设立的很多岗位，创业公司里可能也没有，所以产品经理还需要做客户服务、用户研究、数据分析或销售之类的工作，直到公司变得更大。在创业公司，不太可能是 CEO 让产品经理去承担这些职责，通常都是产品经理先注意到有不足，然后开始动手解决。

作为公司的第一个产品经理，引入工作流程是一项有难度的工作。创业公司的工程师喜欢尽快交付代码，对成本开销非常警惕，所以在添加敏捷之类的流程时一定要小心。产品经理需要确保每个人都能发言，并且要让他们明白添加时间线和里程碑的原因。但实际情况并不一定总这么糟。很多时候，团队只有在清楚需要更多流程时才引入产品经理，所以团队可能已经准备好了。

尽管创业公司内部资源比较少，但他们一般比大公司更愿意寻求外部的帮助。创业公司的产品经理经常会跟其他创业公司的产品经理接触，分享建议。很多创业公司都跟其他创业公司建立了联系，所以他们会形成真正的社区。比如说，一家做 B2B 的创业公司可能会通过他们的投资人跟其他创业公司的联系来寻找它的早期客户。

另一个比较大的差别是关于职业发展的。在大公司，产品经理经常会考虑晋升问题，并会沿着职业阶梯向上爬。而对于创业公司而言，产品经理则更偏重于促成公司整体的成功：营收与盈利、IPO，或者被收购。这意味着产品经理需要从公司整体出发去考虑问题，要让整个公司成功，而不能单纯考虑自身团队的得失。

1. Asana

Asana 是一家面向企业的创业公司，构建现代化的生产力软件。不像传统的企业公司，Asana

的业务模型是自底向上的：终端用户把 Asana 引入团队，而不是 IT 部门。也就是说产品经理专注于构建人们喜欢用的软件。产品团队的产品经理有 Facebook、谷歌和微软的背景。

Asana 的产品经理 Jennifer 之前是微软的产品经理，讲述了从大公司到创业公司是什么样的：

> 因为我们是扁平结构的小公司，所以当我加入 Asana 时，我感觉自己就像是连升了三级。除了管理产品中内容丰富的领域，我还影响了公司的其他领域，比如产品策略、开发流程和企业文化。
>
> 比如说，我能影响总体日程和公司的工作节奏以及产品的设计原则，衡量并控制公司在创建新特性、优化已有特性，以及改善内部效率等方面所投入的资源。
>
> 正是因为我们的团队规模较小并且看重个人的持续成长，我才能有机会从事产品不同方面的工作。有时我需要深入考量某一特性的 UI，而有时则需要对某一方面进行更多的数据分析。

2. Foursquare

Foursquare 是一种基于位置的社交网络服务，帮人们充分利用他们所在的位置，比如，搜索吃甜点的理想场所，了解一场正在流行的中央公园演唱会，或是寻找刚下飞机的外地朋友。

Foursquare 的产品经理主要来自谷歌。Foursquare 的产品经理 Noah 阐述了由此产生的影响。

> 这肯定影响到了我们的文化。不过我们没有按照谷歌的做法全盘照搬，而是选取与我们公司规模相对应的特点。随着时间的推移，再相应地加以改变。
>
> 公司只有 30 个人时，个人的 OKR（目标和关键结果）都是多余的，大家一定要想着“我要尽我所能，帮公司活下去并得到发展”。
>
> 现在我们有 150 个人，个人 OKR 看起来仍然像是没必要的开销，但我们已经开始用团队一级的 OKR 跟踪公司的度量目标了。
>
> 跟谷歌相比，实际上我们编写的规范更多。我觉得是因为这里的项目推进得太快。也就是说我们要尽快、尽早地解决最大和最有争议的问题。我们没时间等着它们后来跳出来，那样会导致延迟。规范也不是构建特性所有部分所需的完整指南，只是记录了团队作出的那些不是显而易见的决定，供人们在以后参考。

3. Dropbox

Dropbox 帮上亿人安全保管他们的重要文件，并且让用户可以通过多种设备访问自己的文件，使他们用自己的手机轻松上传照片、分享文件，或者就某个项目进行协作。该产品团队中的项目经理来自谷歌、Facebook、微软和 Zynga，并且有几个人在加入 Dropbox 之前是创业公司的创始人。

Dropbox 的产品经理 Matt 讲起了在那里工作的感觉：

在一家有 200 人的创业公司工作跟在大公司工作很不一样。在谷歌作产品经理时，我完全专注于如何构建一款优秀的产品，以及如何带领我的团队走向成功。但在 Dropbox 这样的公司，我不仅对公司文化的形成产生了更多影响，还影响了产品的构建方式，以及人们对产品经理本身职责的探讨。我还得以跟创始人密切合作，明确公司愿景及各种事务的优先级，顺带着，还要带出一支优秀的团队。

我们理想中的产品经理不仅要有技术背景、异常敏锐的产品直觉，还要关注细节。我们的创始人 Arash，哪怕产品少了一个像素他都能注意到，所以我们真的希望产品经理也能“吹毛求疵”。

我比较喜欢的一段工作经历是在加入公司一周后，我们要推出一个新的安全特性（两步验证）。我们组织了一支出色的团队，从想法到设计再到推出，在 5 周内把桌面、移动端 App、网站和开发者 API 全都做完了。看到一个小团队能这么神速，真是让人非常满意。

4. 优步

优步推进了世界移动的方式。借助他们的移动应用，租车人和司机实现了无缝连接，他们让城市更加通畅，为租车人提供了更多可能，也给司机带来了更多业务。产品团队成员的个人经历都十分广泛，有来自投行和管理咨询业的，也有来自创业公司的，还有在亚马逊和谷歌这种大公司工作过的。

优步的产品经理 Mina 谈到了在优步做产品经理的卓越之处。

在优步做产品经理就像掀起一辆豪车的引擎盖。作为用户，你只管享受驾驶就行了，不用考虑让汽车顺畅运转的机械结构。但作为产品经理，你必须掀开引擎盖，重新调整它的复杂性，让它工作得更好。

用户可能会专注于我们的移动应用，这也正是我们想让他们做的。而我们则要关注整个系统，帮你找车，帮我们的司机做生意。因为我们不仅和工程师及设计人员紧密合作，还有我们的运营团队，我们很愿意在团队中承担这种独一无二的职责。

经过日复一日的发展，特别能反映我们文化的是全球化视野。仅仅过了三年，我们的业务就拓展到了 40 多个城市，并且这个数量每周都在增长，也就是说我们做的每个产品都要适用于国际用户。我们还非常支持度假式工作（workcation）——优步出钱，让你跟你的同事到一个国外的风景圣地开发优秀产品。比如说，截止到 2013 年的新年，墨尔本、迈阿密、斯德哥尔摩和巴厘岛都有我们的团队。

5. Airbnb

Airbnb 是一个社区市场，人们可以在上面发布、发掘和预订世界各地的独特房源。借助于

Airbnb，人们能在全球 192 个国家 34 000 多个城市挑选到任何价位的房源，享受到独一无二的旅行体验。

Airbnb 的产品经理被称作制作人，就像电影的制片人一样。Airbnb 的一位制作人解释道：

> 比如说一部电影。制片人要跟创作团队、演员、摄像和灯光合作，合力完成它。在杀青那天，你要和导演一起为影片负责。

制作人团队非常多元化——既有来自大型科技公司的产品经理，也有来自小型创业公司的人，更有来自酒店行业的人。这种多元化的人才结构让 Airbnb 构建出了一种非常独特的文化。这种文化基本上是设计驱动的，并且非常强调客户研究，被称为“用户洞察力”（User Insights）。

真正成就 Airbnb 独特性的是它的双方市场（主客双方），以及丰富的线下经验。Airbnb 的制作人会考虑到线下“帧”（再次引用了电影行业的术语）。

Airbnb 的制作人 Thomas 跟我们分享了一个例子：

> 你可能会觉得 Airbnb 影响不了主人招待客人时友善和专业的程度，因为那是发生在线下的旅行体验。实际上我们可以做很多事情，帮助主人超出客人的预期，帮他预见到客人的需求，从而促成有意义的旅行以及独特的体验，并且让主客之间建立起长期的联系。

第4章 积累合适的经验

如果你问面试官他们想找什么样的人做产品经理，他们一般会说希望找到能完成任务的聪明人。

这种愿望会在产品经理的招聘启示上反映出来。招聘启示上所列的要求往往会非常详细，但最终可以归结为两点：

- 你能作出正确可靠的决定吗？
- 你能扫清一切潜在障碍，最终交付出一款优秀的产品吗？

在考虑需要积累什么样的经验时，你应该重点关注这两个标准。

比如说，只要有可能，就尽量坚持跟踪项目的进程，直到项目结束。一定要弄明白所做的事情以及相应行为所带来的后果。不仅要确定最终是否成功，还要弄明白成功（或失败）的指标是什么。一定要考虑何种因素导致了成功（或失败）。有时候失败没关系，但你得知道为什么失败。

4.1 应届毕业生

毕业之后就能做产品经理还是挺不错的。很多大公司都有校园招聘计划，能把应届毕业生培养成顶级的产品经理也是他们引以为傲的事情。从很多方面来看，应届毕业生进入产品管理岗位比有经验的应聘者更容易。

应届毕业生产品经理计划每年接到大批申请，但只能接受其中的一小部分人。如果你想脱颖而出，就需要展示出很强的技术技能，还要有卓越的客户关注能力和产品设计技能。一般来说，只列出你技术类课程的优秀成绩是远远不够的，因为那样很可能会把你的简历归到软件工程师那一堆里。

如果你是在校生，想让自己更容易地当上产品经理，请考虑以下这几点。

- **主修计算机科学，或者至少要辅修计算机科学或密切相关的专业** 校招一般只会考虑有技术背景的应聘者。
- **选修双学位，特别是经济或商业领域的** 计算机科学/经济双学位是产品经理很常见的背景，因为它表明了你对软件技术和商业两方面都有兴趣。要学习分析技能和熟练使用统计数据，经济学是很好的学科。像心理学、哲学、认知科学、人机交互（HCI）或社会学也都是跟产品管理相关的学科。
- **参加团体项目的课程** 团体项目课程是获取领导技能并开始积累相关经验的好办法。注意团队动态以及团队所完成的挑战。在回答面试中的行为问题时，你完全可以把那些经验当作趣闻来谈一谈；
- **担任领导角色** 可以是任何组织的领导，运动队、俱乐部、班级，等等。要带领一支工程师团队，产品经理需要有很强的领导技能。如果你能从零开始筹划一个项目或领导一支团队，比如发起一个新的俱乐部、一次全校范围的竞赛，或者一次课外活动等，那就更棒了，你可以借此来展示自己的积极性；
- **开始一个课余项目** 让你脱颖而出最好的办法是做一个课余项目，比如一个移动应用，这样你就有机会展示自己的关注客户和设计产品的能力。如果你自己没有掌握所需的技术，可以学习，也可以雇一些开发人员帮你做，或者跟懂技术的朋友搭档；
- **去做产品经理或软件工程师的实习生** 边干边学是最好的方法，做实习生就会得到这样的机会。

应该做开发者还是产品经理实习生?

虽然应聘产品经理不要求由担任工程师的经历，但是如果你热爱编码并且希望提升自己的技术能力，工程师实习生能为你提供一个很棒的机会，来从另一个角度看待产品经理和开发者之间的关系。许多工程师都喜欢与曾经担任过工程师的产品经理共事。

重点在于，要展示一些编程之外的技能。能编程固然很好（实际上经常也是必要条件），但那还不够。一定要想办法展示自己的领导力、商业技能和积极性。

4.2 充分利用人才招聘会

如果你看起来不像传统的产品经理人选，那么人才招聘会将是一种你入行的好途径——尤其针对一些比较小的公司。当你跟招聘会展位里的人聊起来，并递上自己的简历时，他们会记录下对你的印象。那些记录决定了是约你面试，还是忽略你的简历。

这里有些小技巧，可以让你充分利用人才招聘会这个机会。

- 研究一下招聘会上的公司，找出你最感兴趣的那些，看看他们是否要招聘产品经理。
- 挑选最好的话题。想一想那些能让你脱颖而出并且能让招聘方信服的内容。这些内容可能是你参加过的有难度的班级项目，也可能你有丰富的助教经验。任何可以展示你积极向上的事情（出于兴趣做过 iPhone 应用，在你的城市里发起过新的竞赛，等等）都是很好的谈资。
- 练习一下如何进行一种简单的自我介绍，重点放在如何使其便于引出后续话题。可以想象你正走向一个展位，跟招聘人员打招呼，很自然地介绍自己所取得的成就。
- 想好你要向那家公司职员问的问题。你可能没办法提前知道前来招聘的是产品经理、工程师，还是招聘专员，所以一定要准备好合适的问题，以应对各种状况。
- 在招聘会上，要注意展台前是否拥挤。这可能是了解一家小公司或是不熟悉的组织的一个绝佳时机。如果展台前人不是太多，他们可能愿意多花时间跟你交流，你就有更多的时间来说服他们，让他们觉得你是合适的人选。
- 在你走到展台前时，要让他们知道你对产品经理职位感兴趣，问一下跟谁谈最合适。他们可能将你介绍给一位产品经理，或者至少是可以评估产品经理的人。
- 友善地跟负责招聘的员工交谈。记住，在招聘会上，他们也是在尽量地向你推销他们的公司，所以提问题是完全合理的。告诉他们你为什么对产品经理这个职位感兴趣，为什么觉得自己适合做产品经理。另外，还要让他们知道你对他们公司有兴趣。
- 不要事先还没跟展台的人聊过就交出你的简历。招聘会的独特优势在于，可以跟公司的员工亲身交谈，所以如果你那么干，就错过了这种机会。

在你想要保持自己专业形象的同时，也不妨尝试做点不同的事情。有些个性和激情挺好的！如果你做过一些特别酷的东西，可以给展台员工看些图片，或者是特短的 demo。

4.3 你需要读 MBA 吗

产品管理不需要 MBA，在有些创业公司 MBA 甚至对你不利。不过，比较关注业务的团队可能会把 MBA 当成宝，有些公司，比如亚马逊，明显倾向于招 MBA。

Arjun 是在微软做过产品经理之后才去读的 MBA。他注意到设计良好的产品并不总能赢得市场时，就决定去读商学院。他说：“我发现光有好的设计还不够，这里还有其他的因素在左右，并且我想知道这些因素到底是什么。念了商学院，我才意识到关于商业方面的问题其实挺容易解决的，一张草稿纸就够了。”

这种分析产品的方式改变了 Arjun 对产品特性和实现优先级的认识。他不再靠猜来定位用户需求，而是开始考虑他想要追求的是哪些指标。

如果你对产品管理感兴趣，那么下面的这些建议可以让你充分利用在商学院里求学的时间。

- ❑ 找机会启动一些事情。通过启动某种项目、加入俱乐部或开发些什么东西，以此来获取一定的经验。这能帮你避开读 MBA 所带来的最大缺陷：只知道发号施令，但根本不知道实际流程。
- ❑ 选择基于项目的课程，可以在那里验证你的想法。这样你上的课就是一箭双雕，并且还有额外的好处：有一队 MBA 在帮你。
- ❑ 练习产品设计，跟同学分享你的模型，并根据他们的反馈进行反复修改。
- ❑ 选择相关的课程，比如创业、营销或消费者行为。

跟 Arjun 的经历比较类似，Chris 也是做了工程专案经理（EPM）后读了 MBA。在谈到读商学院所提供的诸多好处时，他说道："MBA 可以带给你经验、社会关系、想法、资源、种子资金和合作伙伴。商学院确实能提供很多资源。"

你在网上的形象重要吗？

现在人们经常会在网上出现。从 Twitter 到 Quora 再到个人网站，有很多种途径可以把你的想法传达出去。那么你是否应该花很多精力建自己的网站或发微博呢？

你在网上的形象会在两个地方发挥作用：猎头和创业公司。猎头经常会在网上搜索合格的候选人，可能会根据你在网上发的内容找到你。如果你希望受到关注，在网上保持活跃会得到回报的。

领英（LinkedIn）是最有价值的专业招聘网站。有一种办法可以优化你的档案：找找你感兴趣的那些产品经理，看看他们的档案长什么样，他们有哪些杰出之处。你也可以付费开通一个高级账号，看看人们当前是如何查询你的档案的，并据此进行改进。

如果你应聘的是创业公司，可能也会发现有些面试官会在网上搜索你的资料，注意到你发在网上的内容。

然而，大公司的面试官几乎不会事先搜索你的信息，看你的个人网站或发过的帖子。如果你确实有个很棒的网站，可以把它放在你的简历里，或者试着在面试过程中提一下它。但如果没有，也不要担心。

现任的产品经理

如果你现在就是一名产品经理，想跳槽到另一家公司的相同岗位上，那事情就简单多了。如果你曾经在知名的软件公司做过产品经理，那种经验通常足以让你赢得另一次面试的机会。但你的经验也还是有提升空间的，以下是我们的一些建议。

- 推出产品！判断一名产品经理的能力最重要的办法就是看他推出过什么产品。如果你的团队即将推出产品，但还差一点，那你最好等到产品完全推出后再开始求职。同样，如果你所在团队的交付周期很长，你就应该考虑换到一个产品推出频率更高的团队，以便能够见证产品的整个生命周期。
- 申请专利。尽管对软件专利的争议非常多，但很多公司还是把它们看作是必需的。如果你的公司会申请专利，一定要确保为你自己的创新申请专利。一个有专利的应用肯定能让你的简历脱颖而出。
- 尽量完善自己的技能。如果你在产品设计上真的很强，那么可以再尝试学一下数据分析。如果你解决过非常艰深的技术问题，那么何不再花些时间做做用户研究呢？

如果你试图从一个不太著名的公司跳到一家著名的公司，这些建议可能会特别有用。

4.4　为什么技术经验至关重要

很多产品经理岗位都要求有计算机科学学位。乍一看这种要求，可能会觉得莫名其妙：编程并不是产品经理的常规工作，那么那些公司为什么不放宽这个要求，找一些真正精通产品管理核心技能的人呢？

简单说就是：很多没有计算机科学背景的人都在努力跟工程师们建立密切的工作关系。

如果产品经理不能融入工程师队伍并赢得他们的尊敬，那么所有那些优秀的产品管理技能就都派不上用场。产品管理是一项没有权力的领导工作，做出优秀作品的唯一办法就是用你的愿景带领团队起航。

不过，计算机科学学位并不能像魔法一样让你跟工程师形成良好的关系，况且没有技术背景也有可能成为一名伟大的产品经理。其实，许多公司这样做，还在于他们认为，技术背景决定了候选人是否拥有产品经理所必需的一些品质。

- **能够与工程师形成相互尊重的关系**　公司总是会让招聘的产品经理加入现有的工程师团队，所以他们不愿意招那种不能跟工程师团队相处，或者得不到团队尊重的人。
- **对工期长短有很好的直觉**　优秀的产品经理能理解他所用的技术框架，帮助团队制定任务优先级，从而能够平衡工程时间和产品带给消费者的价值。
- **斗志旺盛并能自我激励**　伟大的产品经理以行动为导向，热衷于取得成果。他们会尽量打理好自己能做的事情，不管是收集数据还是修改产品中的拼写错误。这会把开发人员从繁琐的任务中解放出来，让他们做更有价值的工作。

如果你没有这种技术背景（或者在你的简历中没体现出来），尽量找个办法培养并展示这些技能。

4.5 从工程师变成产品经理

因为技术经验如此重要，所以工程师去做产品管理有很大优势。作为工程师，你真的需要知道做一个产品需要什么，以及采取的各种权衡方法都会有什么影响。你自己合作过的可能就有你想要模仿的优秀产品经理，也有那种缺乏必需技能的产品经理。

4.5.1 以客户为中心

从工程师转为产品经理时，最重要的是培养自己以客户为中心的能力。工程师和开发人员一般都掌握了许多其他技能，但能否做到以客户为中心是产品经理优劣的决定性因素。这就是说不仅要想出很酷的创意，更要坚持不懈地考虑到目标受众，他们的希望与梦想，他们的需求，以及他们跟你和你公司的其他人有什么样的差别。

要做到以客户为中心，一种办法是跟使用你现有产品的客户交谈。请产品经理或销售团队的人拜访客户时带上你，他们一般会喜欢带上工程师的。当客户提出需要某些功能，或告诉你他们需要什么时，看看能否再深挖几层，了解一下深层的动机。你基本上是在对客户的需求做一个根本原因分析。

如果你不能直接拜访客户，有时可以查阅，甚至志愿帮客服回答顾客提交的问题单。站在客户的立场上，解决他们的问题，可以帮你形成对客户的同理心。

还有一种办法可以培养你以客户为中心的能力，为你做的功能特性编写像故事一样的用户情景。在这些情景中，把你自己放到客户的位置上，想象一下这些特性如何融入他们今后的生活中。可能看起来很傻，但当你引入了客户思维方式中的细节部分，构建出的产品就能更好地融入客户的生活。比如说，Sally 今天真的会为了更新她的 Flash 播放器而打开电脑吗？很可能不会，所以更新应该在后台安静地完成。

如果你是后台工程师，可能不会跟不同职能组的人有太多交互。试着把依赖你的工作成果的人当作客户。对他们做用户研究，并考虑他们的用例。

在你为应聘产品经理作准备时，要练习从客户的角度描述产品的功能特性。如果你想让人认真考虑你是否适合这个岗位，那么就应该像个产品经理而不是工程师那样讲话。

4.5.2 敢想

在转型做产品管理时，还需要富有远见和战略性思维。作为工程师，你可能非常注重能做到些什么。在你职业生涯的大部分时间里，你可能一直都在让其他人抛弃不切实际的期望。但作为产品经理，你需要那种直觉，让自己施展想象力，将不可能变成现实。

团队需要能带领他们走进未来的产品经理，做那种从来没人做过的东西。虽然在开发过程中，你有机会缩减范围，但如果你想做一个有影响力的产品，一开始就要敢想敢干。

这听起来可能很疯狂，但对于你在做的任何产品或特性而言，要想着它将如何改变这个世界。如果你觉得很难，这里有些提示。

- 试着将实现的好处跟人类的基本需求联接起来，比如安全感、有益，或自尊。
- 从“如果我有一个魔法棒……”开始头脑风暴。
- 写下实际的反对意见，然后继续前进。
- 找个队友在你的头脑风暴中扮演现实的悲观主义者。
- 把“敢想”写下来让它随时提醒你。
- 用写新闻稿的方式开始你的功能特性规划。

总之，要允许自己做梦。

4.5.3 拥抱沟通中有说服力的元素

很多工程师都习惯于分析式的思考方式。作为工程师，最好是通过数据而不是个人魅力去证明一件事情。但作为产品经理，两种办法都要掌握。

我们愿意把所有同事都想象成完美的逻辑思维生物，但在现实世界中要做成什么事，经常要集合团队并把大家鼓动起来。与其给大家看一个满是令人信服的指标的电子表格，可能还不如说一句：“我已经看过了所有的数值，我真的觉得这次应该赌一把！”

诚信对产品经理来说就像金子一样宝贵。当然，所有主管和工程师看了你的表格也都能得出相同的结论，但他们要你这个产品经理就是为了不自己做那件事。他们想让你做研究并提出方案。消除模糊性并帮助团队向前是你的职责。

你越是确信能给出正确结果，说的话越有说服力，你的威信也就越高。如果最后证明你对了，你就会赢得信誉，将来能在更大的事情上说服团队；如果你错了，那他们下次可能对你就不是那么信任了。

这也不是说你要用个人魅力通吃一切。说服工程师最快的办法一般还是用数据，并且在你建立起自己的威信之前，它可能也是更有效的办法。只是不要忘了你的工具不止一种，作为一名高效的产品经理，你应该把它们全都用上。

4.5.4 产品经理跟工程岗位不同，要未雨绸缪

在你考虑成为一名产品经理时，有一些变化是你能预料到的：你不能再写代码了，你要为很

多决策负责，你会在会议上花费更多时间。做产品经理还有一些不太起眼的工作。

了解这些区别能让你的转变过程更顺利。在进行切换之前，你一定要考虑好做产品经理的不利之处。知道自己更喜欢做工程师后，你才不会想要去做产品经理呢！

1. 工作是无形的

做工程师时，你看到自己编写的代码可以正常运转时就会有成就感。作为产品经理，你的大部分工作都没有这样具体的成果。从开发人员变成产品经理的 Stephen 说："做开发时每天都能感受到自己的成功：我的代码能用，我的构建通过了。作为产品经理，你只能自己寻找成就感：我说服了这个人，我让团队出发了，等等。"

2. 你变成了被批评的焦点

创意可能来自团队中的任何人。作为产品经理，你要负责让这些创意凝结成具体的产品。在某个创意还只是想法时可能所有人都喜欢，但一旦它变得具体了，不管设计得多好，总会有很多反对意见出现。作为产品经理，你要能够建设性地接受批评意见，不被个人情绪左右，能化批判为动力做出更好的产品。

3. 你没有时间把它全做了

工程师经常会在一个功能特性上不断深入，并且会给出他们自己估算的时间。他们可能会把一整周都用在一部分代码上。作为产品经理，你不能总是按照自己的想法安排时间，所以你需要排列优先级。这意味着你有时候必须在工作只做了 30%时就交出去，然后继续前进。

4.5.5 在你的团队中寻找机会

对工程师来说，进入产品管理领域的常规途径是在所在团队里寻找机会。这能说得通，你现在所在的团队尊敬你，并且了解你的能力，而这也是你擅长的领域。

谷歌的 Daniel 在他的团队内发现了做产品经理的机会，于是从工程转岗做了产品管理。"我的团队有一个要经常分析数据的产品经理岗位机会，当时并不是特别清楚谁来做合适，"他告诉我们，"我知道如何处理数据，所以我成了这个人选。因为我懂如何跟人交往，并且之前已经有了良好的人际关系，也有威信，这些对我成为产品经理都有帮助。"

对于转岗到现有团队的产品经理岗位上，这里有些提示。

1. 让大家知道你的兴趣很重要

找出谁是负责招聘下一个产品经理的负责人，跟她聊聊。你的团队现在可能不需要产品经理，但当机会出现时，你肯定想在排名上比较靠前。这样你也有机会了解一下她对你的看法，从而在应该提高的地方做些努力。

2. 重读你之前的绩效考评，重点关注所有潜在的产品经理问题

人们是不是说过你固执？团队是不是在你把东西做完之前一直都不知道你在做什么（可能表明沟通有问题）？想看到你在自己负责的功能特性上表现出更多的领导力吗？马上采取行动确定这些问题，充分运用产品经理应有的力量。

3. 即便没有头衔，也要主动承担一些产品经理的工作

可能会有一些小的功能特性需要给出规范，或者可以帮着做些产品方面的决策。如果团队里有产品经理，你可以帮帮他们；如果没有，你可以主动去承担一些值得做的工作。

4. 取得另一种领导权并协调工作

微软的 Stephen 就是从开发转向产品经理的，他跟我们分享了一个故事："我是团队里的技术领导，跟其他两位开发人员一起做一个多团队的协作项目。我们对另外一个团队的严重依赖是个很大的风险，因此我承担了项目管理的角色。我跟合作团队一起努力把依赖项理顺了。为此我的团队认可了我做项目管理的能力，人们也开始把我当作领导了。后来我跟产品经理主管谈要调到她的团队里时，感觉就像是件很自然的事情。"

5. 考虑如何清晰地宣示从工程向产品管理的转变

如果你还在处理大量工程上的工作，很难成功地成为产品经理，你需要采取些手段快速逃离那些工作。比如在换岗之间留段时间，或者举办某种仪式或聚会宣示这一转变。然后你就可以潜心投入到产品经理这一新岗位上了。

4.5.6　在你擅长的领域找一个专业的产品经理岗位

作为工程师，你可能已经有一些自己擅长的领域了。特别是在那个行业有很多专业知识时，你可以用自己的经验更快地成为那个行业的产品经理。

可能你做过安全和加密产品。这是有深度的技术领域，开发人员的经历自然会给你加分。

另外还有个领域你可能有天然的优势，那就是开发人员使用的产品或平台类的产品。既然你做过开发，应该知道如何为其他开发人员做一个优秀的产品。

走上这条路后，一定要凸显你的技能和经验对团队的价值。不要羞于分享奇闻轶事和对行业的见解。

4.5.7　去商学院

MBA 能给你一个崭新的开始，用商业教育充实你的工程背景。这可以把你变成"完美"的产品经理——虽然这么说有的人未必同意。此外，你还有机会跟那些有前途的商业领袖建立联系，

从头到尾地做项目，还有可能成立一家创业公司（或者只是探索一些创业的想法），成为合格的 MBA 实习生，跟顶级公司的产品经理招聘人员建立联系。从开发人员转来的 MBA 学员，特别是参加了顶级 MBA 课程学习的，特别适合去做产品经理。

然而读 MBA 成本很高，会花掉你很多钱，还要两年的时间，并且因为你要读全日制的 MBA 课程，需要离岗，所以也没有薪水了。这样你的投入很容易超过 25 万美元。因为要在金钱和时间上投入这么多，所以如果你只是单纯地想要得到一份产品经理的工作，这可能并不值得，还有更快、更便宜的办法实现这种转变。

不管怎样，如果你曾经在工程岗位上待过，或者你除了要做产品经理外还有其他原因要去读 MBA，那么这个学位可能会比较有价值。

在踏上 MBA 这条路时，要注意你想怎样关注商业或营销。尽管所有大型科技公司都会聘请有 MBA 学位的人做产品管理，但这些产品经理岗位跟工程师的关联程度是不同的。比如说，微软的产品经理岗位肯定是在营销体系之内的（微软的专案经理更像其他公司的产品经理）。问一问招聘人员，确保你对那个岗位有充分的了解。

4.6 从设计师转向产品经理

设计师很适合转行做产品经理。如果你觉得自己目前对产品的影响力还不够，或者觉得自己的分析技能没有得到充分发挥，那么你也可以去做产品经理。

设计师已经知道要以客户为中心，也能设计出优秀的产品。有些公司的产品经理要做大量的交互设计，所以那些技能还会发挥很大作用。

4.6.1 练习排列优先级

从设计到产品管理最大的改变是要负责确定优先级。作为产品经理，你要为产品的交付负责，也就是说你要避免功能特性推进太缓慢，在从工程那里了解到更多成本信息时，还要划定实现的范围。

如果你一直都靠产品经理告诉你何时控制它，那么现在应该开始自己练习这些技能了。有时需要用简单快捷的方式做事，有时需要砍掉一个功能特性让产品的可用性更强。作为设计师，为了提高排列优先级的能力，你可以对你设计的各个部分排列优先级，并跟产品经理讨论它们。看你能否做出相同的决定，如果不能，试着弄清楚是什么原因造成的差异。注意那些因为团队没有时间而被拖延的功能特性，不管是显式的还是隐式的。

有种办法可以有效地磨练你对优先级的感觉，你可以在自己的设计被推出后继续跟踪它们。看看能否跟客户聊一聊，或者看看客服的问题单，以便了解你排的优先级是不是正确。是不是有

人抱怨怎么没有那个被你砍掉的功能特性，或者对你力争保留的东西说三道四？大部分人会发现他们还能砍掉多得多的东西。

4.6.2 增强分析技能

分析技能在产品管理中主要有两种作用：分析你的团队应该做什么，以及如何说服人们去做。作为产品经理，你需要习惯于找到有说服力的数据。那些数据有时来自于产品指标，有时来自于用户研究，有时来自于竞争力分析。

做设计师时，要想办法展示你的分析技能。比如说，你可以通过调查获取数据，从而影响设计。或者可以学学 SQL，用它获取使用情况指标。这些技能对你成为产品经理会有帮助的。

4.6.3 寻找领导机会

在有些公司，设计者很自然地处于领导岗位，而在有些公司你需要找一个能发挥更多领导职能的岗位。你能参与路线图的讨论吗？你可以主持团队会议吗？也许你能找到一个团队可以一起做的项目，把它抛出来，号召人们为之付出努力。

也许你们团队的产品经理现在正好有些焦头烂额。你能帮她分担一些压力吗？也许你可以帮她做一些规格说明，或者试着分析些数据。大多数情况下，产品经理都会乐意看到有人帮忙的。

在没有人给你许可的情况下行使领导职能会让人觉得尴尬。此时你可能需要“在变成领导之前假装自己是领导”。记住旺盛的斗志在产品经理工作中的重要作用。保持旺盛的斗志就是要有丰富的资源，并且能在传统过程不起作用时找到致胜之道。比如说，如果你发现工程师们不愿意动手修复 UI 上的 bug，你可能会想出一个竞赛来激励他们。那种故事真的可以给人留下一个好印象。

从客服转过来

Paul 现在是名高级产品经理，凭着他对用户痛点的理解从技术支持变成了产品经理。

当客户报告说有东西坏了的时候，他总是会问他们当时正要做什么，以便了解客户的真实需求。然后，他填 bug 报告时，会把问题的详细描述包含在内，建议解决问题的可能的办法，提出可能会有帮助的新功能特性。CEO 注意到了这一点，因此让他做了公司的第一位产品经理。

Paul 分享了一些如何从技术支持转向产品经理的建议：“做技术支持时，你一次面对一位客户，解决他们的特定问题并让他们满意。作为产品经理，你必须面对整个用户群体，需要从大局出发考虑和谈论事情。阐明一个问题在更大的体系中是如何工作的。”

4.7 从其他岗位转过来

进入产品管理领域可能很难，但做这份工作的人什么背景的都有。有时诀窍就是坚持。专注于让你的技能适用于产品管理，并且不停地争取进入那些公司。

下面是一些你在应聘产品经理岗位时可以强调的技能。

- **分析** 你当前的工作需要处理数据吗？你是 Excel 高手吗？很多软件公司都在寻找精通数据分析的产品经理，要对指标有感觉，并且能从使用模式中得出见解。
- **以客户为中心** 你的岗位是面向客户的吗？你学会如何把客户反馈转变成行动了吗？那些公司都喜欢能理解客户以及他们需求的产品经理。
- **商业头脑** 你习惯于把商业案例放到一起吗？你知道如何确定一个市场的大小吗？你的经验在作出正确的商业决策方面可能真的很有价值。
- **营销** 你有营销背景吗？你能有效地传达产品的价值吗？营销技能可以帮产品经理设计出有良好市场表现的产品。
- **行业专家** 你对自己所在的行业如何运转有深刻的认识吗？如果有，那你已经有不错的基础去做那个行业的产品经理了。你对行业的理解意味着你可以在短时间内成为高效的产品经理。
- **乐于助人** 你的团队需要得到一些产品管理方面的帮助吗？你有余力承担吗？很多人进入产品管理岗位就是因为当空缺出现时冲了上去。

4

在你想要去做产品经理时，用这些产品经理的技能跟你当前岗位所需要或所展示出来的那些技能比较一下，你就会找到办法来填补经验和技能的不足。

调动关系网

在寻求进入产品管理领域的机会时，不要低估关系网的能力。特别是如果你曾经表现出相关的技能，有曾经一起共事的人能帮你说话，那么对可能给你机会的团队来说会很有说服力。

莎拉从谷歌的合作伙伴技术经理转到 Twitter 去做产品经理，她说："认识人真是太重要了。我是通过认识的人引荐才进去的。原来他们并没有招聘产品经理，因为他们不再需要产品经理了。那些人对我大加赞扬。他们在谷歌跟我有过合作，可以为我做担保。我作为内部工具产品经理加入了 Twitter。加入公司后，我就转到了一个面向消费者的团队。"

4.8 什么算是一个好的副业项目

提升你产品管理岗位竞争力最好的办法之一是做一个副业项目。副业项目让你有机会积累交付产品的经验，提升简历的档次，展示你的技术技能，展示你的产品设计技能，在你面试的时候

可以给你很多谈资。如果你还雇了人来帮你做，可能还可以给你一个展示自己领导技能的机会。

一个好的副业项目有下面这些特质。

- **填补你的经验空缺** 比如说，如果你没有计算机科学的学位，可以通过做个网站或手机 App 来展示你的技术技能。网上有很多免费的在线教程，你可以跟着做一个简单的 App。
- **展示你的技能** 如果你有卓越的视觉设计技能或精明的商业直觉，副业项目是很好的展示方式。
- **有能让你充满激情地讲出来的东西** 你解决的是一个重要问题吗？你是要测试一些有趣的假设吗？你学到新技术了吗？一定要能针对你的项目讲个故事。

如果你不是技术人员，有几个选择。可以聘请朋友帮你实现。可以雇人做。还可以用你的副业项目找出一个有创意的可行性。比如说，你可以设计一个产品，创建一个营销网站来描述它，并邀请人们注册，以便在你可以供货时通知他们。然后在那个产品创意上迭代，看看会对注册率有什么样的影响。

另一个不错的非技术选项是一个设计和可用性项目。在全世界或当地的社区中挑个问题，开始跟人们谈论，观察他们，然后拿出些想法。做一个书面原型并找人对它进行测试——带着它去咖啡店，问问人们是否愿意试试。顶级的设计公司 IDEO，他们的网站上有许多内容谈论以人为中心的设计（Human-Centered Design），对于你了解如何运行这样的项目会有帮助的。

如果你准备提及自己的副业项目，那就要准备好应对面试官就此提出的大量问题，所以对于副业项目的动机一定要诚实，还有你究竟做到了什么程度。为了粉饰自己的技术技能，做一个手机 App 是完全合理的，为了积累推出一个产品的经验也说得通，或者为了尝试一些新的设计范式也合乎情理。实际上，那些都是很好的理由，说明你有学习和实验的热情。

要准备好谈论如果有机会继续做，你将会如何改善那个项目，你为什么要那样选择，还有如果它不成功怎么办。如果你有任何不错的数据指标（用户注册、每用户回报等），这些都很值得讨论。

你应该把自己的副业项目放到简历上，如果你有网站的话，在网站上也放上。如果你的副业项目成功了，很好，但只是做过一些事情也很能说明问题。不要羞于把“不成功”的副业项目挂出来。

第 5 章 职业发展

恭喜你现在已经成为了一名产品经理！跟喜欢的团队一起合作，并设计一款令人期待的产品，这些都是你梦寐以求的，接下来还要做些什么呢？

5.1 职业发展小贴士

努力工作可以让你更加成功，但仅凭努力还远远不够。毕竟，能够推动工作进展的人并不一定投入的时间最多。

同样重要（或者说更重要）的是你努力的方式与目标。在打造自己职业生涯的过程中，以下这些建议会帮你取得更大的成功。

5.1.1 交付优秀的产品

作为一名产品经理，衡量你成功与否最大的标准就是你的产品。这比绘制的操作图、定制的规范或解决的 bug 都要重要，人们对你的认识往往取决于你的产品在市场上的表现。如果客户喜欢你的产品，并且这种良好的口碑还能以病毒式的速度传播，那么就会为你树立起一个良好的形象。

最终，人们不会记得你为了推出产品所付出的那些点滴努力，而只会记得你带领团队推出了一款成功的产品。

5.1.2 发布一些产品，积累经验

从规划到实施，再到推出以及之后的工作，产品的生命周期决定了产品经理需要在不同阶段，掌握不同的产品管理技能，因此如果你想学习和提高自己的管理能力，就需要经历几次完整的产品生产流程。

不同团队，其产品生命周期长度也会有所不同。在产品策划初期，如果能和一支产品推出周期更短的团队进行合作，则可以让你更快地积累相关经验。

5.1.3 成为业界翘楚

新加入一个优秀的团队时，你很容易一有问题就去请教你聪明的队员们，因为他们几乎无所不知。但如果仅仅是把问题抛给别人，你的贡献就不那么明显了。

相反，你真的应该多花点儿时间去研究你的行业和客户，确保自己成为行业翘楚。首先要想一想可以通过哪些研究来真正了解行业：可以去拜访客户，或是组织一次竞争分析。同时，也可以看看数据和指标，或跟销售团队了解相关信息。为此，你应该尽量多跟相关人员沟通，并从团队中了解相关背景。

相信只要做足功课，你就会正真了解你的行业，决策时也会更有信心。

5.1.4 加入一个可以不断学习新技能的团队

Optimizely（以及谷歌早期的）高级产品经理 Chrix Finne 曾说过："资历不外乎是经验，但是有一个窍门，你可以控制自己积累经验的速度。"

如果你之前一直致力于改进成熟的产品，那么你可以考虑加入一个主打推出新产品的团队；而如果你一直从事客户导向的产品，则可以考虑尝试一些业务导向的工作。首先了解自己的技能储备情况，再思考你需要哪种技能，从而加入一个能学到相关技能的团队。

5.1.5 选一个能学到最多东西的公司

在职业生涯的各个阶段，你需要学习不同的东西。选择一个能够为你提供支持的公司，可以促进你的学习。

如果你是新上任的产品经理，就需要学习产品管理方面的基础知识。建议你选一家有几个强力产品经理的公司，以便向他们学习。

跟其他产品经理们紧密合作，是一个了解行业的好办法。在担任一段时间产品经理后，你可能想要挑更重的担子，并学习如何更加独立地工作，此时可以考虑加入一家规模较小的公司。

5.1.6 选择一家不断成长的公司

加入一个不断成长的公司里，随时都会有新机遇，并且你也能够迅速变成资深员工。这也意味着：即便必须加入一个不是自己中意的团队，得到的头衔不是自己想要的，今后也很有可能会迅速得到转变。

“加入一个不断成长的公司，并且研发一款优秀的产品，你就会拥有更多机会。”Twitter 的产品经理莎拉如是说道。她个人最开始做内部工具研发工作，之后转到了面向客户的团队。

5.1.7　选择一个信任你的上司

很多优秀的产品经理都将自己的成功归功于让他们有机会证明自己的上司。选择加入一个团队时，不要只看产品，还要考虑你的上司是谁。

可以经常和公司的其他产品经理沟通，了解谁是最好的上司，而哪个上司要躲着点。一旦找到了好上司，就要努力让他们知道你忠实可靠，并且能够做出优秀的作品。然后，告诉他们你的职业生涯发展规划，并且勇于承担他们交予你的重任。

5.1.8　努力提升工作效率

作为一个产品经理，你所面对的任务会堆积如山，因此你要懂得如何取舍。同时，你还要随时响应你的团队，决不能成为瓶颈，进行合理的时间规划绝对是至关重要的。

如果很难“清空收件箱”（处理完所有邮件），可以尝试一下 GTD 之类的时间管理系统。有时候，对例行工作进行一些小改变就可以让你变得更加高效。随着效率的提升，你会惊喜地发现，每天好像多出来了好几个小时。

5.1.9　如何让自己的角色与公司的发展相结合

产品经理的职业发展通常意味着要拓展工作领域。为实现这一目标，可以从产品的某一功能特性开始做起，接着去接触更大范围的产品管理，直到最终负责整个产品，乃至整个产品系列。

要想遵循这条职业发展道路前进，首先要了解产品链上的各个部分是怎样组合成更大规模的产品架构的。不妨先想一想“如何将我负责的这个特性融入这个产品”，然后再思考一下“如何让这个产品融入整个产品系列”，对这种关联多加考虑，能让你的思路和视野变得更宽广。

5.1.10　帮助团队尽快拿出些实际的东西

大多数团队都会对新来的产品经理心存疑虑：担心工作强度会因此而提高，或是工作进度被拖慢。产品经理所做的大部分幕后工作都打消不了这种担忧，所以同事们可能看不到你所付出的努力。作为团队里新的一员，你可以通过帮助其他成员来打消他们对你的疑虑。

巡视一下整个团队的工作，看看能不能帮某些员工处理一些棘手的问题，或是研究一下那些迟迟没有完成（但真的希望他们已经做了）的工作。这种简单的办法很容易就能让你融入新团队，继而赢得他们的好感。

5.1.11　做一些对团队和公司更为重要的事情

很多公司都会交给新人一些没什么意思的工作，因为这样不用冒太大风险就可以考察一个人的能力。这时候，就需要你在那些没什么意思的工作上做出亮点，证明你可以胜任更重要的工作。然后你可以找到自己认为重要的领域，并作出贡献。

一定要能区分出单调乏味的团队和真的不重要的团队。对产品经理来说，基础建设听起来没什么乐趣可言，但这对公司的发展却至关重要。就基础设施建设而言，产品经理所做的改进能被放大到整个公司，因此这对你的职业发展也相当重要。

5.1.12　接受跨团队或公司范围的任务

在职业发展生涯里，如果想得到晋升，在公司里的知名度就显得很重要。有时只需推出一个对公司很重要的大项目就能得到这种知名度，但是也有其他一些方法能让你出名。

领导并出色完成全公司范围的一个大项目，比如 UI 评审或目标设定，就可以让你获得更多的认可，证明自己是一位优秀的产品经理。另外，你也可以执教培训课程或出席全员会议。当委员会决定是否应该给你升职时，如果他们觉得你的名字如雷贯耳，并且非常欣赏你的工作，局面就会比较乐观。

这种跨团队工作的价值还体现在另一方面，那就是可以帮你和全公司的人建立起关系。有时候在团队之间开展工作比较困难，但如果你已经跟另一个团队的产品经理确立了良好的人际关系，事情的进展就会顺畅得多。

5.1.13　定义并衡量成功

作为一名产品经理，想要真正地脱颖而出，办法之一就是将成功带给团队的价值给予更多的量化处理。根据项目的具体情况，成功可能是指用户的大量增长，营收的增加，或者消费者互动的增加。对于其他项目，也可以定制更加具体的功能特性目标。

确定目标，再把目标传达给团队，继而衡量是否实现了该目标。这样就会清楚何时达成目标；如果没有达成，你也能尽早知道。一旦清楚了自己的目标，就能明确哪些工作是有所裨益的，而哪些工作则不重要，从而能够更好地设定任务优先级。

5.1.14　不要让团队做不重要的工作

有时大公司不能追踪所有正在进行的项目，并且作为唯一的产品经理，你被委派的团队可能真的不能给公司带来太大的贡献。

理想状态下，可以引导团队去做一些更重要的事情。如果不存在这种可能，则可以考虑向你的上司建议解散那个团队（一定要事先做好调查，确保建议会被采纳），或者考虑换一个团队。推出无关紧要的产品会让你很难得到升迁的机会。

5.1.15 不能被动，应该主动出击

作为一名新晋的产品经理，你可能会觉得自己的工作就是交付一些提案、规范、分析报告之类的文档，写好之后，你的工作就算干完了。

然而，那些文档并不是你的工作，它们只是你用来得到结果的工具。要确保工程师按照规范构建功能特性，如果发现他们不能这样做，那么或者重写规范，或者换一种办法来表达你的思路。要考虑某个提案是否能说服你的团队接受；如果不能，就需要再找其他办法来表达你的观点。

5.1.16 证明自己一定能胜任更高一级的工作

大多数有一定规模的公司都有明显的职业发展阶梯，描述了每个级别所需要的技能。这些阶梯看起来如此明显和具体，可能会让人觉得沮丧，更何况还有其他满足当前级别所有需求的产品经理没能晋升呢。

这里有两件事情需要注意。首先，在升级之前，必须确保你能满足下一级别的所有要求。其次，要树立一个好名声，让人知道自己总能胜任下一级别的工作。有时你证明了自己一次，但团队对你仍旧没有信心，担心你不能一直成功。

5.1.17 找导师

产品经理有各种技能。有些人很擅长给团队设立愿景，而有些人则精于数据分析或设计。找到那些你认为在某个领域中确实很强的人，多和他们交流。

应该确保带着具体问题和这些导师接触。可能你想将自己的想法做成一个案例，并找一些真正有说服力的人帮你做提案框架，也可能想让可用性方面的专业人士为你设计的模型把关。明确沟通目的对双方都更有好处。

5.1.18 打造声誉

“对产品经理来说，声誉就像金子一样宝贵。”从工程师成为产品经理的谷歌员工 Daniel 如是说。作为产品经理，你应该赢得所有合作者的尊重，不管他们是产品经理还是其他岗位的员工。尤其要在其他团队领导那里留下良好的声誉，这样他们才会想让自己的人跟你合作。

打造声誉最直接的方式是交付结果。你的队友都想要有好的结果，所以他们自然就会开始猜测你的观点，问很多问题，并提交很多方法与建议。经过一段时间之后，当他们看到你的判断所带来的良好结果，自然就会信任你了。

另外一种打造声誉的方式是注意人们对你的观念，确保给他人留下的印象如你预期。要让大家觉得你聪明、娴熟、干练、可靠，并具有优秀的决策能力。人们可能并不会直接告诉你他们对你的印象，所以有时应该就此问题从上司那里了解一下。

5.2 访谈：费尔南多·德尔加多，雅虎高级产品管理总监

从谷歌产品经理到雅虎资深总监，你是怎么跨度这么大的呢？

在玛丽莎·梅耶加入雅虎任职 CEO 不久之后，我就决定要到雅虎来了。我的信念有了一次很大的跃升。我觉得尽管事情存在很大风险，但还是应该尽早加入，因为这能让我选择一个有吸引力的岗位，也会为我争取到一些筹码。

就大公司而言，如果一直呆在同一家公司，你很难有大的飞跃。因为这些大公司已经设立了明确的晋升机制。有些人即便已经被放在了快车道上，也得按部就班地来。如果公司不那么做，就会有很多不安分的人。

对此我深有感触，在大公司，要想有大的跃升，就得选择跳槽。风险越大，回报也会越大。你可以去创业公司，或者像我一样找一个非常独特的机会：大公司换了新 CEO，开始愿意用新的视角看问题。

这可能会让你觉得，只有每一两年跳一次才会有大的飞跃，但这并不是我想要说的。必须要在某个领域成为专家，而且还得抓住一些机会。如果企业看到你在一家公司从来没呆满过 18 个月，他们就会觉得你在他们那里也一样，所以不愿意冒这个风险。因此，最好在一家公司干几年后再考虑跳槽。

在你的职业生涯里有哪些关键突破点？

到位于苏黎世的谷歌地图工作，对我来说是一个突破点。我在谷歌第一年担任的是 APM（助理产品经理），负责谷歌 Web 搜索质量方面的工作，所以当我被委派到谷歌地图工作时，第一个任务就是负责地图搜索质量。

在 Web 搜索上，有些想法和框架很实用，但在地图搜索上的应用却并不多。那么我想可以把之前在 Web 搜索上学到的一些经验带到地图上，这些观念和技术可都是屡试不爽的。

我曾在一个业界最成功的产品团队（谷歌搜索）中工作过，并能把从中学到的东西带到另一个产品上。鉴于之前已经取得了一些在前沿团队中工作的一手经验，所以这些经历也可以为其他产品的研发提供价值。在我的引导下，谷歌地图团队开始严肃地考虑那些适用的成熟技术。当然也需要具体情况具体分析，做出相应的变动，但最终你会发现，基本理念如出一辙，在开发新产品时，这些技术非常有效。

另一个突破点就是我搬到了苏黎世住。苏黎世办公室的很多人都没有机会亲身体验 Web 搜索如何运作。我引入了一种他们大多数人都没有接触过的新观念，由此就在分部的工程师中树立起了威信。

所以说，假如你对某些事情十分精通，能够接触并掌握它们的运作模式，那么这些经历就会让你不断地成功，并不断推动你的职业生涯。在苏黎世分部，即便新产品的功能不同，我也会自发地将过去的经验融入进去——尤其过去还和顶级团队合作过，那么这种经验就更显得弥足珍贵了。

关于这一点，还有一个类似的例子：Facebook 和 Twitter 在拥有和培养一支逐渐壮大的团队方面都取得了巨大的成功。其中一些文化已经开始对各个创业公司产生了影响，并为他们指明了方向。曾经做过产品拓展项目的产品经理通常有着很多可以借鉴的经验。就用户数而言，尽管从 0 到 500 万跟从 1000 万增长到 5000 万的情况自然有所不同，但很多经验都是普适的。

对于想在自己的职业生涯中有所发展的产品经理，你会给出什么建议？

要知道很多产品经理岗位都会有一个临界点，过了这个临界点之后，做出决策以及团队合作都会容易得多，而且工作进展也会事半功倍。相应地，工程师也基本上不会对你冷嘲热讽或怀疑了，工作环境也开始变得友善起来。

开始在新岗位任职时，应该认真考虑如何到达那个临界点。实现这一目标的难度可能大相径庭。举例来说，现在我正领导雅虎的 iPhone 和 Android 移动应用的研发项目。刚开始整个团队只有两个人，我和一名设计师。后来随着人越来越多，我重新组织了整个团队，从而顺利度过了临界点，随后很多事情都变得更顺利了。

我曾经也在那种似乎根本无法通过临界点的项目组里工作过，随后我就决定换一个新项目做做。比如，在 Android 营销团队里，领导层都很强势，大家互不相让，没有一位明确的最终决策者。这种架构使得人们经常只能妥协，采取错误的折中方式进行决策。每个工程师都亲眼目睹过这种争执，作为产品经理的我很难通过临界点，成为强有力的决策者。

公司 CEO 拉里主张尽量让每个团队里只有一位最高决策者，但这个团队却始终无法确定这个角色。很多工程型领导者都有很强的产品理念，这可能不错，但我也非常固执。在一个争执不休的团队里工作，解决问题时很难避免冲突。

Android 是一个非常成功的产品，并且团队里的工作人员都非常努力地工作。可能是因为团队结构不合理，并不是每个人都能得到相应的回报。这种矛盾在我的大女儿出生时达到最严重的地步。尽管在高强度的环境下努力工作，我却没能得到相应回报。

我知道自己不可能度过那个临界点了，因为其他人在团队里工作的时间都比我长。面对重压，他们自有办法应对，而我却只能带着工作中的沮丧情绪回家。最初加入 Android 团队时，用户有 5000 万，后来增长到了 3.5 亿。这确实值得骄傲——能为这个伟大、成功、呈指数级增长的产品作贡献。但我打乱了工作与生活间的平衡，换来一种前途不明的职业发展道路，所以我决定换个团队。

有了这些经历，我更加意识到必须要找出临界点并知道如何通过它。有些因素在你的控制之内，而有些则是你无法控制的。选一个可以长期工作的团队，并待到比大多数人在那里的时间都长。考虑一下你愿意跟一个产品多长时间，你不能让时间变快，但却可以选一个更可能成为资深成员的团队。

还有一点需要注意：在早期，你应该尽可能地做一名观察者，不要急于改变或打破什么。接触到一个产品时，每个人自然会有自己的认识，觉得需要改变什么，但大多数情况下，已经有了固有的决策文化，并且事情既有的优先级都是有原因的，千万不要操之过急。

首先，你要做的就是了解事情的始终缘由。要让自己变得更好奇，而不是告诉人们要做什么，也不要尝试做决定。为什么会是那种方式？尽量了解历史背景，而不要成为团队中的新独裁者。

还要提醒一点，尽早到达临界点的办法就是把你的价值凸显出来，努力让有些事情缺你不可。通常会出现这样的情况：人们都觉得有些管理性或无聊的工作很有价值，应该有人去做，但就是没人动手。

比如，做一款新闻类产品，需要显示新闻来源的 Logo。目前行业里采用的是手工提取，那么你完全可以自告奋勇地去做这件事。找一种可扩展的方式，比如智能抓取，或者采用亚马逊的土耳其机器人（Amazon Mechanical Turk）等机制来做。使人们免除某些繁杂的管理或无关紧要的工作，但一定要确保这能使产品增值。

这里要给更为资深的产品经理一个忠告：找出你自己的决策体系和原则，并且尽可能地传达它。在这些体系的形成过程中，你身边的同事有时并不能觉察出来。

要让周围的人都清楚你制定决策的缘由脉络，这样他们就明白你的立场。工程师最恨的就是变来变去的主观决定。如果能够形成某种体系和原则，人们就会熟悉你的决策风格和立场。

这里有个例子，移动组织的高级副总 Adam Cahan，对移动应用的动画和过渡有着极其敏锐的感觉。每次让他检测新做的应用，他总会注意到其中一个过渡的不协调性。他并不会限定过渡的做法，但他注意的是，一旦为应用的过渡选了一种模式，比如隐喻风格或通用思维模型，那就一定要坚持到底，努力将其做到完美。

如果选择的过渡有一种深度感，并且会随着应用视图的不断切换越来越强烈，那么这种作法本身并无大碍，但绝对不能在打开设置面板时破坏这个原则。为你的应用选择一种风格，然后全面贯彻它。

有时，当你给 Adam 看一个很棒的动画，他却告诉你别那么做。一开始你可能会有抵触情绪，但是他把自己的原则传达得如此清晰，以至于你不得不相信他这么说有充足的理由。

例如，我通常都会尽量确保这样一条原则：当客户打开一个移动应用时，应确保他们以最直接的方式看到该应用的功能和价值。所以如果添加了一个不太常用的功能或特性，我会避免为其设置欢迎界面和教程。我的原则是：不要预先显示它；你可以引导人们去使用，但不要轻易设置障碍。

经过时间的检验，我们意识到，如果一直坚持某个原则，你就会发现不同于以往的解决方式。所以，我们会为有多样需求的用户提供快捷的使用方式，前提是确保客户对某些功能特性感兴趣时，再给出相应提示和弹出框。我们会耐心等待，只有当客户用非快捷方式使用过某个功能特性并对其足够熟悉之后，关于快捷方式的弹出提示才会有意义。

5.3 访谈：阿什利·卡罗尔，DocuSign 产品管理高级总监

讲讲你的职业发展道路。

本科毕业之后，我加入了摩根大通。在那个时代，对获得经济学位的人而言，这就是一种潮流。那时我在旧金山的办公室工作，最终被分配到了科技、媒体和电讯组。这是一个很不错的机遇，因为我借此机会了解了所有优秀的互联网和软件公司。

在摩根大通主导 Shutterfly 的 IPO 之后，我去 Shutterfly 做了一名业务分析师。在一家运营公司的第一份工作就是分析员，太棒了，因为我可以用报告和分析为各种职能领域提供支持，而且还得以了解他们如何分配自己的时间。我对用户体验上的新特性与变化对客户互动指标（以及最终收益）产生的影响深感兴趣。另外，我还发现自己特别愿意跟工程师和设计师合作。我想要更深入地参与到产品创造过程中，创造一些能够取悦客户的产品。2 年之后，我离职去读 MBA，因为我觉得这是一种从分析转到产品的有效途径。

在进入商学院之前以及在此进修期间，我分别在 oDesk 和亚马逊网络服务进行了实习。通过这段经历，我充分体验到了不同规模、文化和细分行业的公司之间的差异。读完商学院之后，我加入了 Survey Monkey 的产品团队。我开始作为一个专注于增长计划的独立贡献者，比如推出新计划和彻查新用户的产品接受度（user onboarding）情况。最终，我开始管理平台团队，这一团队主要负责的是用户身份识别、计费和企业级特性之类的工作。

几年之后我离开了 SurveyMonkey，到 Optimizely 领导产品，在那里从无到有建立起了产品团队。后来那个岗位跟我最初预期和想要的不太一样，所以半年之后我就离开了。接下来，我变得不再那么盲目了，决定做些咨询方面的工作，以便在正式介入某家公司之前事先了解一下相关团队（以及经理）的实际情况。后来就遇到 DocuSign，我每个星期都有两天的时间在这家公司，接触了几个月后，我觉得这家公司还行，于是就顺利入职，成为了他们的全职员工，负责监督产品的增长情况。鉴于该公司现在所处的上升态势以及产品所具有的优秀品质和口碑，我对这份工作很满意。

在你的职业生涯中有哪些关键的突破性时刻？

在 Shutterfly 做业务分析师时，恰逢当时有产品经理离职，由于我熟悉那条产品线的相关指标，因而就有机会介入其中，积累了一些产品管理的经验。事实证明，这些工作经历十分关键。从商学院毕业之后，正是基于这些经验，我才能够为产品发布和 UX 初步决策等方面提供一些实例。

关于 SurveyMonkey，总感觉有说不完的故事。能在那里工作简直太幸运了。在员工培养机制、个人发展空间及支持等方面，领导层都做得很出色。当我面对一个新项目或职责时，在毫无头绪的情况下，老板和团队能无私地帮我，助我一臂之力，并且还不断让我担负起新的职责，使我能够满怀激情地投入新任务中。

我能担当起现在这个角色，很大程度上有赖于机缘巧合。当时，DocuSign 正在组建产品团队，恰好我就在这时被介绍加入进来。他们在企业级软件方面的人员阵容很强大，而产品营销团队的员工们则更注重考虑客户和用户体验。所以，我的职责所要解决的都是之前做得很顺手的事。另外，在此过程中，还不断出现新的机遇和挑战，能让人逐渐成长。

对于那些想在自己的职业生涯中有所发展的产品经理，你有哪些建议？

加入一家正在经历（或者更好是即将经历）高速增长的公司。业务的发展速度会超过团队的扩张速度，这样你就有更多的机会，去承担更重要的责任。风险投资界可能是这方面的好资源；他们对人才（你！）的重视程度丝毫不亚于你对工作机会的重视程度，所以这是很棒的共生关系。

选择共事的人是重中之重。同事是你最好的学习对象。并且因为技术上的事情可能会产生很大的压力，所以一定要考虑他们的为人。跟之前曾合作过的人一起工作是审查一个人最好的方式。很明显，在你的职业生涯刚开始时会很困难，但经过足够的努力，就可能会找到第二或第三级连接。另一种建议就是先通过咨询来了解一下具体的情况，类似传统的“先试后买”。

能让人产生信任感的管理者和执行领导团队是很重要的。在你的职业生涯中，尽早在创业公司中谋求更高的职位很有吸引力，但我真的很感激在经验丰富的执行层下面工作的那段时间。即便职业生涯发展到这个位置，我也更愿意为那些能够给我授业解惑的人工作，而不是仅仅位居高

层的人。

这听起来是陈词滥调，但事实就是这样。好雇主会对员工进行一番官方和非官方的调查，所以一定要成为那个你想成为的人。这不仅意味着你要努力工作，处事机敏，而且还要诚实、谦逊，要能真诚地关心团队。我就是想跟具备那些品质的人合作。

5.4　访谈：布兰登·布雷，微软集团首席集团专案经理

讲讲你的职业发展道路。

我一直在做项目管理方面的工作。最开始是在微软 Office 的 Outlook 项目组做了两个暑期的实习生。大学毕业后，我加入了微软的 Visual Studio 团队，也就是开发 C++和 C#的团队。

我是从 C++编译器后端开始着手，所以它有很强的技术性。有些人觉得专案管理（Program Management）只是前端，但我要跟 AMD 和英特尔等所有芯片厂商打交道，这些经历十分有趣。

做了一段时间后，我转到了 C++编译器的其他方面，包括为 C++ CLI（通用语言基础结构）设计了语言。我游历世界各地，跟 ECMA、ISO 以及推动该语言的所有公司一起合作。

然后我决定拓宽自己的经验。我接任了一个项目管理的岗位，同时成了一名负责人，主要负责交付 Visual Studio 2008。这仍然是一个专案管理的岗位，不同以往的是这次着重围绕发布管理。专案经理通常只会做一点项目管理的工作，大部分时间还是用在设计上，而发布经理则更关注发布标准（exit criteria），以及其他所有让整个发布良好进行的项目管理工作。

在那之后，我到中国待了一年，在上海组建了一支产品经理团队。也就是那时，我意识到大多数人都是通过观察其他产品经理才学会做产品经理的。在上海，当时还没有多少产品经理，所以我必须利用其他办法帮助那些产品经理迅速成长。

接下来，我回到美国，成了.NET 框架上的一名集团专案经理，这是一个二级经理的岗位。我帮着推出 Windows 8，然后转到 C#和 Visual Basic 团队。这个团队用了款新的编译器，但看起来没办法及时整合，我帮它度过了交付驼峰。

近期，我又加入了一个全新的、未发布的硬件项目团队中。为了作出改变，我决定重新做回一名独立贡献者，并且我会努力工作以确保实现一个优秀的开发平台。

在你的职业生涯中有哪些关键的突破性时刻？

编写最终成为 C++和 CLI 标准的语言规范，绝对是我职业生涯里的一个亮点。为了达成这个目标，我当时必须跟来自不同公司的人合作。

这让我认识到把东西写下来的人是有力量的。不同岗位的人，比如测试和开发人员都认为是产品经理做了所有的决定。后来双方互换角色，才发现其实是编写的人真正记录了历史。不是其他，正是你写东西的方式，以及你写得内容定义了历史，这就是书写的力量。

在这之后，对于如何解决问题，我也有了更大的影响和话语权。还有，撰写就是思考。如果你经历了写东西时的细节，就能够考虑到极端情况。

另一个我要谈到的时刻，是在职业生涯初期，你就可以实现的影响力。我在 C++编译器上做的第一件事是 /GS 特性（缓冲区安全检查），这主要是针对缓冲区溢出攻击而研发的。之前这个团队已经研发了一些方式来抵消缓冲器溢出，但却遗漏了很多情况。对此，我就想研究一下怎样让它更好地发挥作用。

之后，我跟微软的所有团队沟通，说服他们使用新的编译器重新构建代码，这并不是一件容易的事，工作量很大，但产品的安全性有了很大的提升。

最后一个关键性突破就是在过去几年里做了一名经理。我非常乐于塑造产品经理的思考方式并培养产品经理团队文化。组建一个真正有技术性的团队很容易遗忘客户的多样性：包括开发人员、最终用户、IT 员工，等等。我们必须真正关注并铭记这么一点：软件产品不是我们的客户，活生生的人才是。我时常提醒团队，要经常考虑用户体验方面的因素。

你有没有做过一些有助于升职的事？

我做实习生那段经历帮我迅速从产品经理 1 提升到了产品经理 2。正是通过那段实习，我才学会了如何做一名专案经理。大概在成为一名负责人和项目经理的同时，我也被提拔进入了资深岗位。我曾经把想成为负责人的想法写了下来，之后就有团队找到我，请我做他们的负责人，因为他们需要有人来做那份工作。

述职之后，我采用了一个所有部门都适用的开发流程——我们称之为基础和原则。这是用来检验成功产品所应具备的各项要素的一种体系，包括安全性、性能、可靠性、全球通用性、隐私、合规性，等等。这个流程能确保达成以上所有目标，但事与愿违，情况搞得一团糟，他们要我来贯彻这个流程。

于是，我跟各部门所有的产品团队一起合作。我必须组织、管理 300 多人的团队，才能推动事情的进展。结果，我最终扭转了整个局面。那简直就是在玩火，因为那个项目的范围真的很广。自那以后，我就被提拔到首席的级别。

对于想应聘成为产品经理的人，你会给出哪些建议？

有一个观点，我跟所有人都说过，即产品经理应该成为客户的专家。这也是把产品经理和开

发人员及测试人员区分开的特点之一。举个例子，Excel 的产品经理必须是 Excel 用途方面的专家，这些人可能是金融分析师，或者爱摆弄数字的人。Excel 的开发人员不需要懂数值分析，但产品经理必须要懂。

在我看来，客户就是开发人员。由此你必须真的喜欢客户。如果开发人员是你的客户，你需要理解他们的思维模式，把自己也想象成你的客户。假如你是游戏的产品经理，就应真正理解玩家的心理。如果客户是数据库管理员，同样也应这么对待。

作为面试官，不论如何，我最看重的是应聘者对工作的激情。你对自己的工作有足够的热情吗？会花自己的空闲时间去了解它吗？

我对新手的建议是：一定要找一个你在工作时间之外还有兴趣的事情作为职业来发展。不管是回家做业余的项目，还是去聚会，或是参加会议，它们应该是你完成每天的本职工作后还有兴趣去做的事。如果你有那份激情，还有不断学习的动力，那么其实就已经准备好了。

有时人们会问我："我能否在自己还不了解的领域团队中工作?"我的答案就是：如果你对那个领域感兴趣，并有学习的动力，就能在任何小组中取得成功。当有一个目标摆在面前时，人的学习能力真的会很强。优秀的产品经理对他们团队的效率有巨大的影响，团队成员也会在感召之下，帮助产品经理学习。

另外还要讲一个面试技巧。在我面试时，如果想判断一个人应该是开发人员还是产品经理，我通常会给出一个有不同导向的问题。应聘者可以选择直接开始设计方案并构建算法，或者也可以退一步，聊聊客户是谁，目标是什么，以及如何定义成功的标准。

好的产品经理通常会先着手于前者，即实际操作方面的问题，但很快他们就会意识到，需要退一步聊聊客户。如果没那么做，就说明他们不具备一个产品经理所应具备的基本能力。

如果你正在为了成为产品经理而参加面试，最好看待所有问题时都从"谁是客户"以及"什么是成功"开始，我一直坚持那么做。每当我在交叉路口等红灯时，我都会想想："怎么才能把这件事情做得更好？为谁而做？"这些问题总是会浮现在我的脑海中，你也可以用这种思考方式训练自己。

5.5 访谈：托马斯 · 阿伦德，Airbnb 国际产品主管

讲讲你的职业发展道路。

我在柏林主攻数学和计算机科学。第一份工作是在 IBM。一开始我是一名工程师，当时还没有产品经理这个概念。

我发现自己在工作中最喜欢做的事就是用通俗易懂的语言给客户讲解产品，然后回来向团队其他成员宣传我们的客户都是些什么样的人。我愿意跟用户和潜在用户聊天，并试图找出他们想要什么，但通常第一阶段都是听他们发牢骚。

我花了很长时间才明白什么是产品管理。我乐意干我做的事情，但在我的上司看来，我是个表现欠佳的工程师。在审查我的绩效时，他说："从工程角度来看，你的表现不算上乘，但不管你在做什么，继续做吧。"所以我就继续了，并且当我听说有产品经理这个岗位时，我觉得很振奋，因为我意识到我其实就是产品经理！

我在 SAP 做了几年的产品经理，然后调岗到稍微偏业务方向的岗位上。我跟着 CEO 一起做了几年公司的战略方面的工作。那是很宝贵的学习经历，但我更怀念做产品的日子。

所以我又回到产品和设计领域。11 年后，我学到了很多知识，公司也成长了很多，以至于我的兴趣也转向了 Web 和更小型的公司。

我离职去了谷歌，在那里工作了 5 年。我主要致力于产品国际化方面的工作，并提出国际化应该实现 40 种语言的倡议，后来我在玛丽莎・梅耶的团队里做 iGoogle 项目，之后又在巅峰时期离开。

我在 Mozilla 待了一年，并被 Mozilla 和非盈利的概念所吸引。不过事实证明，我喜欢那种思想，但不喜欢它的执行方式，所以我又去了 Twitter，在那里呆了一年半，做了几个产品。

截至目前，我在 Airbnb 已经有 7 个月了，担任国际部负责人的职务。我的目标是让产品在 192 个国家取得成功。这个岗位不仅仅是做本地化和翻译，还包括在不同国家的战略。我不只是在做一个产品，而是要跟其他产品经理合作，保证在所有国家做出最好的产品。

在你的职业生涯中有哪些关键的突破性时刻？

可以说有很多个时刻造就了今天的我，其中很重要的一点就是跟很多优秀的岗位模型产品经理交谈。加入谷歌时，我跟 Sundar Pichai 和很多之前就在那里的人交谈，他们都是我非常尊敬的优秀产品经理。我不是抄袭他们的风格，而是取其精华，去其糟粕。我从他们那里学到的东西实在太多了。

现在我到了那些需求的另一侧，换做是我跟所有联系我的人会面。我强烈建议那些想要成为产品经理的人建立起自己的人际网络。我们都是通过阅读来学习如何写作的。类似的道理，观察好的产品，看看别人如何做出用户喜爱的产品，然后联系那些公司的员工，询问他们是如何实现的。

另外一个重要时刻是我做设计服务时，比如在 SAP 内部有一支 IDEO 式的设计团队。那是一个很棒的团队。我曾参加过该团队的几次研讨会，它确实让我大吃一惊；在只有一天的研讨会里，

我学到了跟用户换位思考和快速进行原型设计。他们让你明白了用户的真正需求到底是什么，并教你如何仔细观察，而不是急于得出结论。

很多时候，就像是被扔到一个新问题中，比如重新设计一家书店。在几乎没有指导的情况下，你需要走出去跟用户交谈，拍照，回来之后进行合成，制作原型，然后带着原型去用户那里找反馈。

年轻的产品经理可以自己做这些事情。如果你住在山景城，可以去实现一项潜在的任务，比如把使用加州火车售票机变成一次愉快的体验。那么就可以按照以上那些步骤来做：观察，拍照，跟用户交谈，并了解你的客户。

找出你的人物角色——他们可能是每日通勤的上班族，也有可能是游客。然后综合问题。也许显示器上会有眩光。你需要综合问题，排优先级，简化到主题，然后构建原型。你甚至可以制作纸质或纸板原型。之后再回去找那些用户，问问人们是否愿意试试你的原型。你可以独自开展这些工作，当然和其他人一起合作更好。我认识的一个朋友为此开了一个博客，并自己做了很多事情，他都有些上瘾了。

现在，我在一些针对创业公司的咨询委员会做事。我尽量教他们理解谁是他们的用户，以及那些用户为什么会关心他们的产品等问题。在搜索一个问题时，我曾经看到过很多优秀的方案。有一次，我跟一个前来咨询的公司交谈。我问他们，谁是他们的用户，他们说是婚礼策划师、面包师、私人教练、街角的花店。然后我就问他们真正跟多少个那样的客户交谈过，他们说一个也没有。所以我告诉他们，要做的第一件事就是走出去跟他们的用户谈谈，充分认识并了解他们的客户。

对于想在自己的职业生涯中有所发展的产品经理，你会给出什么建议？

要知道自己对什么感兴趣、有激情。

可以借助一些实用的工具来了解你的用户，创建他们对于产品期望的假设，并提出可能的解决方案，也可以尝试一下头脑风暴和角色扮演等方法。

当然，有计算机科学学位或了解相关技术会有帮助。同时，你还需要跟工程师们合作，并赢得他们的信任。如果他们反过来迁就你，你就失败了。

假如我要开发一款新车，我不必是搞研发的工程师，但如果我在自己的闲暇时间里至少能换个轮胎并了解其内在机理，肯定是会有帮助的，并且理想情况下，如果我可以说出所有零部件的名称，并且能动手换一些就更好了。

了解产品的内在原理并对其感兴趣很重要。如果有人对技术完全不感兴趣，那只能说他们并不适合技术领域。如果让你买本书自学一下 Java，装 Eclipse，并构建一个简单的移动 App，你

会有什么样的感觉？如果那让你觉得兴奋，很好。如果你觉得："我真的必须得那么干吗？"那可能你真的入错行了。

要尽可能多地了解设计和用户研究。一个优秀的产品经理不仅要创造出舒适的用户体验，还要知道如何定义并度量成功。毫无疑问，以用户为中心，有指标支持的决策总是最好的。

5.6 访谈：约翰娜·赖特，谷歌副总裁

讲讲你的职业发展道路。

我大学主修的是数学，第一份工作是在一家金融服务行业的小型软件公司做程序员。当时，公司会安排每个应届生跟公司高管进行一对一会谈。在我跟产品经理汤姆会谈时，他对公司发展的清晰规划让我感到震惊。我梦想着自己有一天也能拥有那种技能——知道自己想做什么。从那时起，我就决定要成为一名产品经理。

我开始为成为一名产品经理做各种基础准备——一开始我做了 QA（质量保证）经理，然后我又做回工程师。后来，另一家做互联网平台创业公司找到我，请我做他们的 QA 经理，但我说我只想做产品经理或程序员，于是我就成了一名产品经理。

互联网泡沫破灭的时候那家公司也倒闭了，我搬到了加州。我和丈夫决定从布鲁克林骑行到洛杉矶，我没想过自己在金融危机时还能再找一份产品经理工作，但我觉得自己还是能拿到一个项目管理合同的。我很幸运，在远足时遇到一位朋友的朋友，她雇了我。6 个月后我在 UCLA 开始读商学院。

对我来说，商学院对我的信誉有很大提升。我的创业公司实在太默默无闻了，我毕业于巴纳德，它的计算机科学专业也没什么名气，人际关系网也不强。商学院可能不适合所有人——斯坦福毕业的人，或是曾在其他公司有声望的人，在信誉上不会有这么大的空缺，可能也不太需要读商学院。但对于我而言，就当时就业市场状况来看，商学院的经历还是很有意义的。

从商学院毕业后，我去谷歌做了产品经理，为此我付出了艰辛的努力。有那么一段时间，我的日程安排里有一个周期性的提醒："联系希拉里"，这位女士是一名在谷歌工作的巴纳德校友，是我从巴纳德校友数据库中找到的。最终，由于我不断能够收到其他一些公司提供给我的工作机会，希拉里就可以轻松地将我引荐给谷歌招聘人员了。

我曾加入过谷歌的很多团队，一开始是 Ads UI，后来有被调到搜索团队。在搜索团队中，我推出了通用搜索、即时搜索和知识图谱。接下去，当我准备好迎接新的挑战后，又去了 Android 团队，现在我是那里负责搜索助手项目的副总。

在你的职业生涯中有哪些关键的突破性时刻？

我作为产品经理的第一份工作真的是个突破。我曾在一家小公司工作过，这段经历确实可以给产品经理下一个定义，因为我就是那里唯一的产品经理。我得到那份工作的方式很有趣：当我还是 QA 经理的时候，我跟这家公司的几个创始人一起用餐，我对自己的工作感到由衷的兴奋，并且对于团队的设置和运转也有着自己的一套完整理念。因为我对工作所表现出的强烈热情，所以他们决定雇佣我。

后来到了谷歌，最关键的职业突破肯定是开发并推出了通用搜索。这是公司的重要项目，之前也没有人成功过，所以刚一推出通用搜索，我就被升为主管。

在早期的职业生涯中，担任 QA 经理的经历为我后来的职业生涯奠定了基础。它给了我考虑问题的角度：如何使用产品，如何了解你的用户，如何了解客户的心理。做程序员的经历对我了解程序员的工作过程也很有帮助。事实上，在软件开发过程中，任一岗位的工作经历都会对成为产品经理有很大帮助。

对于想成为产品经理并在自己的职业生涯中有所发展的人，你会给出什么建议？

做些有意思的事情，做些重要的事情。我认为职业发展的答案不止一个。我一直在试图寻找有趣的，并且能让我的大脑全天投入的工作。我职业生涯最大的突破也是我为公司作出最大贡献的时候。

还有一些其他忠告。

比如向人们传达你在做什么。在大公司，取得领导的信任和支持十分重要，因为他们有权分配资源。在做通用搜索时，我们首先要做的就是了解用户。为此，我们要事先准备一个 demo，这对我们的工作也十分重要，因为产品就是我们工作的最佳证明。

绝不放弃。找工作时，要乐于跟很多人交谈并且不要怕被拒绝。我得到谷歌工作机会也不容易。在我的 MBA 班里，我是最后一个得到实习机会的。所以，振作起来，这些都可以逆转。

找一个信任你的老板。在谷歌，对我来说有个信任我的上司真的很重要。有玛丽莎·梅耶做我的经理真的很棒——她给了我很多机会。在我怀孕 9 个月时，她建议我把我们的搜索策略展示给高管。这对我的职业生涯来说非常重要，因为正是此举让我成为人们心目中规划搜索策略的人。而这都要感谢玛丽莎。

怎样成为一名优秀的产品经理？

我曾经合作过的每个产品经理都有独特的个性，这都是很棒的经历。他们都有十分优秀的品质：精力充沛、懂得如何鼓励他人、过硬的技术能力，或者是拥有良好的交际能力，等等。

也就是说，所有优秀的产品经理都是目标导向的。他们能把事情做成，专注，能分清主次。大多数谷歌初级产品经理们犯的一个错误（其他大公司可能也一样）就是参加了太多会议，而且还认为自己已经完成了工作。优秀的产品经理知道怎样实现目标，而效率差的产品经理则受日程安排所限，什么都做不了。

最后，我想说的是优秀的产品经理关心用户体验。他们会使用自己研发的产品，而不是完成产品规范之后就不了了之。他们会持续关注产品的实用性，有始有终地做好每件事情。

5.7 访谈：丽莎·科斯托娃·绪方，Bright.com 的产品副总

讲讲你的职业发展道路

我转到产品管理的背景和传统的背景不太一样。我在宾夕法尼亚大学沃顿商学院取得了国际关系和金融的双学位，毕业之后又去了华尔街几家投资公司工作。

我首先做了 5 年的私募股权和风险资本投资。回想起来，那段时间培养的技能在我的产品管理职业生涯中显得极具价值：评估产品和团队，找出值得投入时间与资源的对象，并计算投资回报。这不仅能够很好地训练一个人的批判性思维，而且还能培养出时常考虑各种权衡决策和机会成本的习惯。另外，这段经历也为发展我的沟通能力打下了良好的实践基础，使我学会了如何将各种猜测简洁清晰地传达给不同受众。

2007 年，我去读了哈佛商学院（HBS）。在那段时间，我逐渐明确了自己职业生涯的真正目标。我一直都觉得，尽管我发现投资非常激发智力，但是我却十分渴望能够参与到产品和服务的创造过程中去——我要亲自动手。

我在读商学院的第一年就订婚了。因为我的未婚夫的工作深深地扎根于旧金山湾区，所以我知道自己将来还会回到旧金山谋求职业发展。Facebook、Twitter 和 LinkedIn 之类的互联网公司的爆炸性发展让我深深着迷。我花了很多时间研究他们，跟他们在一些领域项目上进行合作，并对这些领域产生浓厚的兴趣。

在 HBS 上学的第一和第二年之间，我去了谷歌实习。现在的谷歌对产品经理岗位有了严格的教育要求，并且还明确要求有计算机科学的学位。然而当时我很幸运，加入了在线销售和运营小组做实习。实际上这个岗位跟项目经理更接近，但我很感谢它；它让我有机会接触到这家公司和这个领域，让我踏入了技术运营工作的大门。

因为 2008 年夏末的金融危机，谷歌的 MBA 实习生都没拿到工作录取通知。但是，很多人却因祸得福，我们在寻求职业发展机会时更加灵活和有创造性。有些同学开了自己的公司；其他一些同学在没有拿到录取通知的情况下毕业了。

我是后面这群人中的一员。我想我会搬出湾区，到那种没到 MBA 校园来招聘，但急需帮助的创业和成长型公司去寻找机会。

因为我的背景跟很多成熟公司产品管理岗位的传统要求不一致，所以找工作的过程就像坐过山车一样——我只做过几个月的技术，并且大型组织里的人们倾向于把我转向产品营销经理的岗位上。

过了一段时间，一些朋友向我推荐了 Zynga。这家公司那时候还在急速发展：《开心农场》（*Farmville*）游戏刚刚上线，总经理 Mark Pincus 非常推崇有商业背景的产品经理。这家公司亟需帮助，而且他们要找的是有潜力的人，而不是有经验的。我抓住了这个机会。经过两天充满艰辛的面试，我被录用了。

接下来是几个月的熬夜和埋头工作。那是我能想象到最艰苦的新手训练营体验，有很多产品经理在几个月之内退出了。但我知道为了了解产品经理究竟是怎么一回事儿，我必须坚持，并且我做到了。

我在 Zynga 度过了几乎 4 年的时光，却仿佛经过了 7 到 8 年。人们将在 Zynga 的一年习惯称为“狗年”，因为我们推出新游戏、特性和产品的频率是独一无二的。

截至离开 Zynga 之前，我都一直管理着一个十几人的团队，并且我们负责的是 Zynga 最重要的跨游戏通道和 Facebook 产品。我喜欢我的团队，并且已经跟他们培养了良好的关系，一些人甚至教我如何编码。

尽管我喜欢自己的团队，但我同时也清醒地知道 Zynga 在 Facebook 上的游戏已经没有多大发展了，更何况那是一个正在下滑的市场。我开始寻求新的机会，当时我意识到在产品管理和任何职业生涯中，真正重要的东西是人际关系。我在 Zynga 几个不同团队曾合作过的一些工程师同事邀请我去他们所在的新公司。我加入了两个前同事所在的 Bright，负责领导他们的产品团队。

我在 Bright 跟一些优秀并且聪明的人一起共事，从而又学到了很多东西。这家公司的业务是在一个巨大市场（工作招聘）中的大数据领域，有着振奋人心的巨大机遇。除了几个消费者产品，我还顺便负责一个企业产品，这又是全新的体验（特别是它要跟销售团队和他们的销售流程对接时）。

你觉得是什么让你如此成功呢？

开放的心态，强烈的好奇心，以及不断学习的欲望。不要害怕进入自己不太了解的领域——你必须习惯于迅速进入一个新的，甚至有些吓人的领域。你需要做一个完美主义者，并且要活到老学到老。关键是提问，保持好奇并向你的团队学习。

我总是把工程师、设计师和团队的其他成员当作平等的合作伙伴，他们会解释并教给我他们那个领域的专业知识。我的团队向我证实了编程并不可怕。它建立在逻辑之上，并且并不像我之前所认为的像数学那么复杂。

好奇心和对产品的激情也非常重要，跟培养对客户的同理心一样——如果产品经理不关心产品，会以很多种不同的方式在产品的品质中体现出来。

作为产品经理，你还必须能够跨越多个层次和受众。就像原本使用不同的语言（人类的或计算机的）的人，将消息以一种人们都能理解语言传递出去。你需要让你的团队为你们做的东西兴奋不已，让他们作出贡献，拥有他们的想法，还要转达和管理沟通过程，包括跟管理、销售、法务等等关心产品不同方面的部门的沟通。

你要跟团队成员真正进行沟通与合作，使大家能够满怀激情去工作。你需要成为那种在幕后推动进程的人，快速解决问题的人，毫无畏惧迅速进行决策的人，而不能一整天都独自地在安静里角落里待着。

这也就意味着，作为产品经理，你必须认识到你永远都不可能成为最好的市场营销人员、工程师，或是销售人员。对于这些领域的内容，你都需要足够熟练和精通，但更重要的是要能靠良好的感觉把各方面人员都融合到产品中。我感觉自己就像是乐队指挥。

MBA对产品经理有什么样的价值？

读 MBA 就是为了 MBA 学位，在科技行业里没什么价值，至少在硅谷是这样。有些行业肯定对它是有要求的，比如管理咨询，但科技行业不在其中。

话虽如此，但顶级学校的 MBA 也不是毫无价值，它的价值就在于能够建立起人际网络；我的很多同学都参与了业务和产品创造过程，大多是各个公司的创始人、风险投资人或其他关键角色。所以我自己就能融入到一个非常强大的人际网络中，无论我接下来决定在自己的职业生涯里做什么，都能创造和组合出很多机会。

我喜欢 MBA 的另一个原因是能有两年丰富多彩、忙碌充实的学习时光，并且能在一个非常激发智慧、思维开放的环境中接触各种公司、行业和人。我喜欢学习，在 HBS 度过了自己人生中成长最迅速的阶段。但我在自己的日常工作和交流中不会宣扬我的 MBA 学位或在 HBS 的求学经历。在我工作的地方，在我做的事情中，重要的是为公司的成功作出贡献的能力。我的有些同事，人都非常聪明，但他们甚至都没有读过大学，所以说，教育背景并不意味着太多东西。

MBA 可能会对你的发展有好处，但你需要认真考虑，你想从中得到什么，以及你为什么要为此投入时间和金钱。

对于想成为产品经理并在自己的职业生涯中有所发展的产品经理，你会给出什么建议？

保持好奇心，每天都要努力学些新东西。跟你身边的人学习——营销人员、工程师、销售人员、QA 的伙计们，等等。没有好奇心和开放的心态，你就会变得四处提防，内心苦闷，甚至痛恨你自己的产品经理生活。

不要追求品牌。不要仅仅因为那是一家时下热门的公司而加入。很有趣，在 Zynga 的初期，很多想要应聘成为产品经理的人都是饥渴的、斗志旺盛、仿佛有某种障碍的那种人。后来，随着 Zynga 在 2010 到 2011 年间成为最炙手可热的首次公开招股之前的公司，很多人都想要应聘成为他们的产品经理，因为他们听说这家公司很火爆，并且猎头喜欢有 Zynga 工作经历的产品经理。

相反，要想想什么能让你觉得开心。考虑一下自己的风格：是分析型还是技术型，是专注于设计，还是富于创意？如何将自己的风格跟组织文化相融合？公司产品能与你产生共鸣吗？你乐于为公司吸引的那类客户服务吗？你知道自己要跟你面试过程中遇到的人长期合作吗？他们正在解决的问题使你觉得兴奋吗？

心态要开放。产品经理不是一个明显的、有严格定义的岗位。很多公司都在没有产品经理的情况下发展了很长时间，有些还根本就没有这个岗位。这里没有预先定义好的一组按钮和拉杆让你操作，你必须觉得兴奋、受到驱使，有上进心，并且富有同情心。你必须能够参与进行，在有需要的地方施以援手——那样才能在创业公司中确立你作为产品经理的职责。

最后，要能在机会出现时把握住它。它可能跟你想得不一样，但如果它能把你引进门，让你离创建产品更近一步，抓住它。要充满斗志，并且相信它最终总能够实现。

第 6 章 面试内幕

6.1 谷歌

网上有很多与谷歌面试流程和文化有关的谣传。有些是绘声绘色地讲述谷歌面试官试图恐吓应聘者的“恐怖”故事，有些是则引述据说是某人的朋友的表哥的同学的室友被问到的荒谬问题。

实际上，真实情况要比那些传说温和得多。

谷歌的面试流程跟其他公司都差不多。一开始是两次电话面试，可能会涵盖技术、文化契合程度、策略、分析和产品设计等问题（尽管可能不是全有）。

如果你通过了上述那些面试，可能会被邀请参加一整天的现场面试。可能还要在那里吃午饭，经历一次午餐面试，这次面试一般不会给你反馈，面试官只是坐在那里提问。

每位面试官都会有各自的任务，分别负责评估你的技术技能、产品技能以及分析技能。

每个面试官都会单独对你进行评估。谷歌对面试官的要求相当严格，在他们全部提交反馈之前，禁止任何面试官相互交流对面试者的反馈结果。也就是说如果你在前面的面试中表现不佳，不用担心那会影响后面的面试官对你的看法。同时，在面试中遇到“容易”或“困难”的问题，也和你的最终评定结果没什么关系。

面试过后，助理产品经理（APM）应聘者会被要求写一篇文章。这篇文章往往侧重业务策略的某个领域。文章要写得简洁、切中要点，不能写成优美、描述性的散文，因为那并不是优秀的商务作品。当然，还要检查下有没有拼写或语法错误。

产品经理应聘者会由一个特定的团队面试，但很多面试官都不是来自那个团队的。APM 应聘者不用由特定团队面试，因为 APM 是要轮岗的。

6.1.1 怎么决定的

要录取谁并不是由面试官决定的。面试官只负责写下对面试的反馈意见，然后交给招聘委员会来决定。

每个面试官会在 1.0 到 4.0 给你打分。但面试委员会作决定时并不是只看这个数值，他们还会考虑你的面试官是不是“苛刻的打分者”。

一般来说，你的面试平均分至少应该达到 3.0，外加一个强烈支持者。因为，即便所有的面试官都推荐雇用你，但如果他们都不觉得你很突出，你仍有可能被拒。

招聘委员会是由你的同行（其他产品经理）、经理和招聘人员组成的。面试官一般不在招聘委员会里，如果确实出现了这种情况，也纯属偶然。

招聘委员会给出录用/拒绝的建议（很少会在下一阶段被推翻）。如果招聘委员会建议录用你，那么你的资料会被传给薪酬委员会，然后到执行委员会，最终确定你的录用结果。

谷歌的招聘流程环节很多，所以他们需要几个星期才能作出决定。

6.1.2 重点关注

谷歌不会像其他公司那样关注行为问题。实际上大多数面试官事先都不会通读过你的简历，他们更喜欢在面试过程中直接测试你的技能。

在每次面试中都要尽量插入几个关键点（恰当的地方），因为面试官可能不会那么深入地探测你的背景。你业余时间所做的优秀项目，加上在分析问题时的出色表现，可能就会让你得到亟需的“强烈录用建议”。

另外，招聘委员会不能跟面试官直接对话。他们必须完全依据面试官的记录作出决定。如果你有某方面的背景想要传达给面试委员会，可以把它穿插在多次面试中。那样可以极大提高面试委员会了解它的几率。

谷歌喜欢就他们自己的产品提问：你喜欢哪个？你会把哪个做成不一样的？要准备好谈论某些谷歌产品的细节。

谷歌还会问很多估算问题和技术问题。一定要温习好你的量化技能和技术技能。如果要求你在白板上写点代码，也不要感到吃惊。

估算问题经常会涉及的谷歌产品的某些方面，其中一大领域是广告。要特别注意案例研究章节中跟广告相关的问题。

6.1.3 助理产品经理

经验不足（不满两年）的应聘者通常会应聘助理产品经理的岗位。这是一个轮岗计划，所以对团队融入的要求明显没那么重要，不过面试流程和产品经理岗位大致一样。

6.2 微软

微软的面试流程可能是所有公司中最简单直接的。实际上，其他很多公司的面试流程看起来都是从微软面试流程衍生出来的：把招聘委员会换成招聘经理，就这么直接。

在微软，应聘通常会从一次或两次的电话筛选面试开始。如果你是大学生，面试可能就在你的学校里进行。

有时会由招聘人员做最初的筛选面试，但也不要因此放松警惕，招聘人员也不是那么好糊弄的。

有工作经验的应聘者也会被安排一次技术筛选面试。

通过电话筛选面试后，应聘者要到雷德蒙德接受一整天的现场面试。面试通常是由一个特定的团队进行的。团队与团队之间的要求可能会不一样，所以特别适合某个团队的应聘者在另一个团队那里可能会显得很差劲。

如果你那天最后能由招聘经理或级别更高的人面试，这是好现象。一般表明你已经通过了技术上的面试，他们只是考察你的文化契合程度，或能否融入微软。也有可能是他们在梳理入围的关键人选，这次面试被称为“酌情”面试。

6.2.1 怎么决定的

面试之后，团队的面试官们会通过 Email 或在会议中讨论你的表现。他们作出决定，并把结果反馈给招聘人员，如果必要的话，还会同时给招聘人员一个录取信息包。

应聘者有时会发现他还没离开微软园区就收到了录取通知。这看起来可能让人觉得不可思议，但要记住：面试官们要做的只是提供他们的反馈意见，并有可能会迅速讨论一下。如果那天恰好所有面试官都可以参与讨论，决定可能会非常迅速。

但如果你没有这么好的运气也不用发愁。决定的快慢取决于面试官们的日程，还有等待未参与面试人员的时间，而不是你的表现。

6.2.2 重点关注

微软特别喜欢行为问题和产品设计问题。

在产品设计问题中，要注意细节，并且一定要问试探性的问题。微软面试官一般会乐于测试你如何处理模糊性。比如说，他们可能会要求你设计一支笔，但并不提及那是给宇航员用的笔。他们期望你在解决问题之前先问很多问题来了解客户。

6.2.3 额外的好处

现在你对微软的招聘已经有了大概的认识，但我们还应该再给你一条建议：注意大家对微软泛泛的认识，特别是涉及招聘时。你的朋友在微软的体验和你即将面临的经历可能没有什么共性。

微软的团队大部分是独立招聘的。一个团队可能想要了解一下应聘者是否具备深入的技术技能，因此会要求你写一些伪代码；其他团队则可能想要测试你的设计技能。什么情况都可能会发生。

6.3 Facebook

Facebook 是建立在黑客文化之上的，这也体现在他们的招聘文化中。Facebook 喜欢有创业活力的产品经理。

跟其他公司一样，产品经理应聘者的面试也是从一或两个电话面试开始的。电话面试经常会问到行为问题，可能还会问你对 Facebook 的热情是从何而来的。要回答好这个问题，可能要讲一些对 Facebook 有真实体验的轶事，或者表现出你想要跟聪明人一起合作做大事的渴望。

如果你在电话面试中表现良好，会被邀请参加 4 到 5 次的现场面试。每次面试都会有相应的特点。

- **技术和逻辑**　在这个环节中，会问你一些量化问题，特别是围绕指标和实施实验。如果你有技术背景，可能还会问你跟代码相关的问题（尽管 Facebook 最近放松了这方面的需求）。
- **设计**　这其中通常会有典型的产品设计问题。此外，如果你做过什么东西，他们可能会要求看一看。还要考虑一下你喜欢什么服务和应用，以及你为什么喜欢它们。
- **未来主义者**　比如像“电视将来会变成什么样”之类的问题，能够表明你对未来的推理能力。你不能只是谈论未来的状况；还要考虑出现什么样的根本性改变才会让未来朝那个方向发展，以及那些改变会产生哪些影响。你要会讲故事。
- **大师**　对于要求丰富经验的岗位，他们会询问你的核心优势。同时，他们会检查你对自己的感觉，以便了解你是否知道自己擅长做什么。

被招进 Facebook 的入门级产品经理通常不会加入特定的团队，而是作为自由人，哪里需要就去哪儿。

经验丰富的产品经理主要是为特定的团队招聘的，但 Facebook 内心深处还是希望这些应聘者能够成为万金油。面试官通常包括一些团队成员和一些非团队的成员。

6.3.1 怎么决定的

录取与否不是由 Facebook 的面试官直接决定的。而是提交书面反馈给招聘委员会，这个委员会由一般员工、经理和招聘人员组成。

一般而言，面试过程中会包含几项评估。招聘委员会需要看到你在所有这些方面的强劲表现，才能决定是否招募你。

6.3.2 重点关注

Facebook 一般会要求产品经理应聘者编写代码。他们知道你可能很长时间没写过代码了，也会考虑到那种情况。所以，代码不是重点，他们只是想找一个能像软件工程师那样思考问题的人。你知道将一个问题分解为几步吗？差不多是这样吧。他们一般也不指望你能了解数据结构和各种算法的具体细节，但你应该知道像散列表这样的基础知识。

6.4 苹果

有人开玩笑说苹果是个邪教组织，可能某种程度上确实是这样的。苹果确实非常看重文化契合，这在他们的面试中也体现了出来。

总的来说，苹果一般会为特定团队进行招聘，而不是针对整个公司。苹果的面试流程从两次电话面试开始，之后你会被邀请参加现场面试。

一些团队坚持用 4 ~ 5 小时（或差不多时间）由团队成员主持的长时间面试。然而其他团队可能会对你进行多达 12 次的 30 分钟面试。这些团队明显非常看重文化契合程度，因此他们想让你和很多人碰面。

你的面试官会来自各种各样的岗位：其他产品经理、设计师、工程经理、高管（比如产品经理的初级副总裁）。大多数情况下，会有一名招聘经理在午餐时面试你，但如果他比较忙，也可能会找人替他。

在这一整天的时间里，有些面试官可能会彼此确认一下，看看他们的流程是否正确。

6.4.1 怎么决定的

面试之后，招聘经理和团队成员会一起开会作出决定。

6.4.2 重点关注

苹果的信条是有激情的员工才是好员工，所以他们想找对公司和它的产品有激情的人。你会被问到很多为什么想在苹果工作的问题。好好准备准备吧。同样，你应该非常了解苹果的产品。准备好描述你喜欢它们什么，以及你认为哪些地方还有待改进。

6.5 亚马逊

亚马逊的应聘者先要接受两次短暂的电话筛选面试。过程只有 30 分钟，并且一般不会深入测试你的技能水平，主要想了解你的背景，以便决定是否应该邀请你参加现场面试。

现场面试由 4 到 6 名面试人员组成，每场面试大概持续一个小时。

面试官会看你跟亚马逊的 14 条领力导准则的匹配程度（参见附录中的“亚马逊领导力准则”），每个面试官都会涉及两三条准则。如果某个面试官觉得自己在某条准则上的准备工作不充分，可能会请其他面试官跟进。

在这 14 条准则中，把事情搞定的能力（“崇尚行动”及“达成业绩”）和“顾客至尚”特别重要。

你的面试官中会有一名“抬杠者”。抬杠者是一位来自其他团队的特殊面试官。这名面试官的任务是“抬杠”，并判断你的水平是否能超过亚马逊 50%的产品经理。通常你很容易把他从你的面试官中找出来：那个来自于另一个团队的人。

抬杠者经常也是那个给你最多挑战的人。比如说，他可能会测试“有骨气”的价值。在恭敬地表达了不同意后，你能继续坚持自己的观点吗？如果在你没有给出令他满意的答案之前，这位面试官继续打破砂锅问到底，请不要觉得吃惊。

当然，可能你还要接受招聘经理的面试。

6.5.1 怎么决定的

面试后，你的面试官会开会讨论你的表现。抬杠者为面试流程负责人并拥有否决权，招聘经理也有否决权；毕竟那是他的团队，也就是说你必须同时打动招聘经理和抬杠者（理想情况下还要打动其他所有人）。

6.5.2 重点关注

尽管像亚马逊 Web 服务这种更偏向技术的团队可能会希望你具备一定的技术能力，但亚马逊的大多数团队一般不会过于关注技术能力，公司更关注的是你的商业技能和背景。

亚马逊所有领导力准则都很重要，但其中“顾客至尚”这一条尤为重要。在有疑问时，应选择去作对客户正确的决定（即便不是正确的“商业”决定）。

亚马逊会问很多专门针对价格的问题，所以你一定要想想亚马逊的不同产品（比如亚马逊金牌会员计划）是如何定价的，想想你能给它们带来哪些改变。

亚马逊面试官喜欢钻研你的简历。比如，简历上说你做的功能把效能提升了 30%，那你最好能给出证据。你要作好准备，准确地调整用词，证明是如何做到的，以及你如何测量它的影响。在这一方面，含糊其词是行不通的。

最后，这些领导力原则不是开玩笑的。如果你注意到了，并且对这些领导力原则有良好的认识，你可能会判断出某位面试官真正要问的问题，然后直接解决它。更好的是你可以对此作出准备，针对某条领导力原则审视你的简历。

6.6　雅虎

雅虎既招产品经理，也招产品经理助理，两者的面试流程大体上是一样的，但也有几处差别。

产品经理和产品经理助理应聘者的面试都是从一或两次的电话筛选面试开始的。成功通过电话面试的应聘者会被邀请去参加一整天的现场面试。

在现场面试时，产品经理应聘者会由来自多个岗位和级别的人面试。至少应该有三个相同级别或更高级别的产品经理同行，加上至少一个来自不同团队的人进行面试。你的面试官中还应该包括一个招聘经理。

在 APM 的面试中，并不是由特定团队对招聘者进行面试。在作出录取通知并被接受后，你会被问及自己喜欢的团队，然后在正式报道前一到两周内通知你所在的团队。

6.6.1　怎么决定的

面试之后，每个面试官会提交书面反馈给招聘经理，然后被归纳到一个文件包里。如果这个团队觉得应聘者可以胜任，录取信息包会被发给招聘委员会，然后给高管做最终批准。

而 APM 的面试中，录取决定是由 APM 指导委员会作出的，然后由高管审查并最终确定。

不管哪种情况，雅虎要找的都是有激情，精力充沛，能把事情搞定和能推出产品的人。

6.6.2　重点关注

雅虎寻找的是技术功底深厚的产品经理，所以你应该想办法证明你扎实的技术功底。你需要

展示自己跟工程师沟通的能力，很少会被要求编码。

你应该还会碰到产品和分析问题，尽量要有一个框架和一个特定的视点。

6.7 Twitter

Twitter 的招聘流程从招聘经理的通用电话筛选面试开始，然后把你和特定的团队做匹配。在被邀请参加现场面试之前，你应该会收到一到两次的电话筛选面试。

在现场面试时，你可能要接受多达 7 次，每次 45 分钟的面试。你的面试官应该有同行（其他产品经理），还有可能会有跟你一起合作的人，比如工程经理、技术负责人，或者来自客服团队的人。

6.7.1 怎么决定的

每个招聘经理的做事风格都会稍有不同，然而 Twitter 一般只在遇到高水平人才时才会发出录取通知，他们想要能给团队带来新事物的人。

6.7.2 重点关注

Twitter 十分看重那种提前做了功课并且喜爱 Twitter 的人。只是因为它的名声而申请加入 Twitter（像很多人一样）是没有用的。你应该真的用过他们的技术，并且真正了解它。你应该是 Twitter 的重度用户才行，而不能只是偶尔用一下。你觉得什么东西真的很酷？它是如何工作的？如果有问题你会做什么？你应该痴迷于为用户创建卓越的体验。

Twitter 还很看重那些能够应对变化的人，因为 Twitter 增长非常迅速。你应该愿意担任多种角色，能承担较大的工作压力，并且有卓越的人际关系技能，因此行为问题非常重要。

Twitter 的产品经理们通常不会被问到编码问题，但他们可能会被问到从技术上如何设计一个产品。你应该掌握预加载和即时计算之类的概念。

6.8 Dropbox

Drew（合伙人和 CEO）会告诉每个新招进来的人，他们的首要任务就是招募其他有天分的人。因此，几乎一半的新人都是员工推荐的，招聘团队也非常积极的从现任产品经理那里挖掘候选人资源。

尽管面试流程仍在进化，但产品经理候选人的特征已经定义的很好了。Dropbox 寻找既有技术背景，也有创始人经历的人，或者在成熟公司作为产品经理作出重大成就的人。

一旦入围，你就要接受其他产品经理的两次电话筛选面试，他们会针对你喜欢的产品和可能的改进问一些典型的产品经理问题。

如果你被邀请参加现场面试，一般会有产品经理、工程师和产品设计师对你进行 4 次面试。产品经理会问产品问题，工程师是技术方面的，产品设计师可能会要求你在白板上设计一个新的产品工作流程。就在那一天，或之后很快，Arash（合伙人及 CEO）也会面试你，他会问你跟 Dropbox 相关的产品问题，还会考察你的文化契合程度。

根据当前需求和候选人的背景，产品经理可能会被选派到特定团队，或者在之后不久再跳到其他团队中的万金油。

6.8.1　怎么决定的

面试后，面试官会一起听取报告，然后作出决定。招聘人员在那一整天都会跟面试官保持沟通，所以如果你没有通过最初的面试，可能没有机会跟 Arash 面对面交流。Dropbox 的门槛很高，并且他们愿意花时间寻找合适的人选。

6.8.2　重点关注

“关注细节”是 Dropbox 的工程价值观之一，这也同样适用于产品经理。回答问题时要准备好，考虑到产品的所有边界情况和设计细节。文化契合也极其重要，但他们不会问任何特殊的问题来考察它。

Dropbox 有非常集中的系列产品，所以要全部熟悉它们，还要认真考虑如果你是那里的产品经理会做些什么。

第 7 章 简历

你能参加面试并不是因为你的经验，而是你如何在简历中呈现那些经验。如果简历做得很烂，哪怕你是这个世界上最佳的人选，也不可能得到面试机会。毕竟招人的公司无从知道你是不是最佳人选。

但实际上很多优秀应聘者的简历确实做得不行。他们对自己的经验缺乏认识，被职业咨询师过时的建议搞糊涂了，没有具体说明他们做过的项目，或者陈述用词宏大空洞，对简历筛选者来说没有任何意义。

糟糕的简历对任何工作来说都是个问题，对产品管理岗位更是如此。沟通是重要的产品经理技能，而简历能清楚地展示出你的沟通能力。一个不能用一种清晰、简洁和有效的方式表达自己技能和成就的产品经理多少让人有点儿担心。所以跟其他岗位相比，简历的质量更能影响招聘者对你的判断。

7.1 15 秒法则

你知道简历是什么，但你真的明白怎么用吗？（等等！不要跳过这一节！为了写出优秀的简历，你真的需要知道这些！）

没人会仔细阅读你的简历，只会粗略地看看。筛选简历的人大概只会用 15 秒（甚至更少）来决定是否邀请你来参加面试。

这决定了简历的指导原则。一份简历应该针对 15 秒的扫视进行优化。

先记着这一条吧。我们会不断地重复这条原则。

7.2　法则

每条法则都有例外，但它们能被称为法则是有原因的。如果你觉得这些法则在你这里行不通，那请你慎重处理。

法则 1：越短越好

想象一下，我要竭尽所能向你充分展示自己，但你的注意力只能持续 15 秒。我是给你一份 300 页的自传，还是只有一段的缩减版？

300 页的版本会有很多信息，但那都没用。15 秒的时间只够你读完第一段。你在那么短的时间里能知道我是在哪里出生的就不错了。尽管我提供了更多信息，但实际上你对我了解的要更少。

一份长长的简历也一样。它把你最好的内容混到那些不太重要的信息里去了，让看简历的人很难形成对你的整体印象。

最好是只突出重点。

法则的实现

首要法则是如果你的工作经验不足 10 年，则把简历限制在 1 页以内。如果超过 10 年，可以做到 1 页半或者 2 页，特别是当你做过很多不同工作时。

你可能会说自己没办法把做过的事情都放到 1 页里。对，确实不行，但你可以把其中最重要的事情放到 1 页中。你必须更精简，但这是好事，说明你在突出重点。

当你觉得应给曾经就任的某个特别岗位多加描述时，请问问自己，这个岗位为何是最重要的：是因为那时你实际上是程序员，还是因为你帮公司削减了成本，还只是因为那家公司的名字。

只关注重点，别管其他的。

另外，如果你的简历只是有几行跑到下一页去了，那就找个办法缩减你的简历。从一份只有一点内容跑到第二页的简历，简历筛选者可以看出你糟糕的优先级排列能力。

法则 2：罗列要点即可，别用大段文本

很多人都是靠“越重要的事儿越要多说”这条法则活着的。这条法则有它的道理，但用在简历上却不合适。文本块越长，筛选简历的人就越不可能去看它。

文本块（即超过 3 行的要点或段落）一般是不会有人看的。一定要短。

法则的实现

通读简历，把所有超过（包括）三行的文本精简一下。此外，还应确保至多有 50%的要点描述超过两行，也即至少应有一半的要点只有一行，剩下的才是占到两行。

根据具体情况，你可能要斟酌所用的词句，或者砍掉一些细节。你的工作所产生的影响一般都比工作的细节更重要，所以略去那些细节也没关系。

此外，如果一个要点中只有几个词语跑到了下一行，也要删减一下，否则就是在浪费空间。

法则 3：工作成果，而不是岗位职责

人们不关心你应该做什么，他们想知道的是你做过什么。

岗位职责是你应该做什么。罗列这些内容只是泛泛地指出了你在工作当中应当做什么。并不能明确你是否真的施加了影响。此外，你的职责一般相当明确。一般而言，大家都知道产品经理或软件开发人员在公司里应该做什么。

相反，重点应该是你的工作成果，应该向筛选简历的人证明你的影响力。

看一下这两条要点之间的差别。

- **面向职责** 为亚马逊 S3 设计功能特性，监督软件工程师和测试人员开发这些功能特性。
- **面向工作成果** 设计了 SS Frontline 功能，管理它的开发，领导它与其他 3 个产品的集成工作，从而为公司额外增加了一千万美元的收入。

第一条要点泛泛介绍了你是做什么的，看过的人从中并没有感觉到“你因为__而成功了。”你的简历应该更像第二条，只有那样才能展示你的成功之处。

法则的实现

看你用的是否是现在时态就能判断出你罗列的是不是岗位职责。你很难用现在时态表达曾经取得的成就。

但千万不要以为注重成就的简历就是用过去时态表述。毕竟把前面那个例子中的要点换成过去时也不能把它变成真正的工作成果。

要用实实在在的方式列出你的影响。专注于影响本身；侧重于“是什么”，而不是“怎么做到的”（尽管两者都很重要）。

尽可能量化你的成就。帮公司赚了多少钱？帮团队节省了多少时间？把客户保留率提高了多少？估算也行。

如果你现在刚好在做简历，下面这些问题也许能帮你从头做起：

- ❑ 最值得你骄傲的 5 件事是什么？
- ❑ 你的团队会对你做过的最有影响的 5 件事说什么？

这些问题的答案应该能构成你简历中的要点。

你的具体工作成果和职位头衔一般应该可以清晰地传达出你的工作职责。但如果你觉得有必要解释一下自己的职责，那就让它紧跟着职位头衔，用不同的字体把它和真正的工作成果分开。

法则 4：找一个好模板

每隔几个月，一些网站或博客就会发布一堆“迷人的”简历设计模板，毫无疑问，很多找工作的人都会照着那些模板做自己的简历。这些简历使用信息图表样式的图表，或模仿 Amazon.com 首页或 iOS 首屏的风格制作简历。

这些简历确实漂亮，可能也呈现出了某种程度上的创造性，甚至还吸引了一些人的注意。但除非你是因为华丽的设计而得到媒体关注的少数几个幸运儿之一（或者你应聘的是设计岗位），否则那样的模板一般对你没什么好处。

很多招聘经理都不喜欢这些图形化的简历，因为很难从中充分了解到你的相关情况。信息没能以一种清晰的方式呈现出来，还占用了很多额外的空间。

好的简历模板不会让你的朋友发出惊叹。可能不会太华丽或特别有创造性。但它能完成任务，也就是说能让你得到面试的机会。

法则的实现

一份好的简历既紧凑又能快速展示出你的优点。

要按下面这几条标准找模板。

- ❑ **两或三栏，一栏放公司名称，其他的放职位头衔**　这些信息应该很容易就能看到，尤其是你的简历上是一家顶级的公司时。地址和日期的重要性要小得多。这些信息要有，但不能分散简历筛选者的注意力。
- ❑ **没有专门放标题的左边栏**　很多简历模板会用页面的左侧放“就业”和“教育”之类的标题。这看起来很漂亮，但也会浪费 20%的可用空间。
- ❑ **限定文本的样式**　太多的字体、大小、边框和颜色都会让人分心。
- ❑ **合理使用空格**　太多空格会浪费空间。太少又会让你的简历看起来不舒服，还说明你不善于排列优先级。
- ❑ **要点**　文本块看起来漂亮（特别是在图形化简历上），但会被忽略。

有很多符合这些标准的简历模板。在 www. crackingthepminterview.com 就有些样本。

法则 5：不要跳过最好的东西

按道理说这太显而易见了。在简历上，你当然不应该对最好的东西有所保留！

然而实际上很多应聘者都会忽略这个最终法则。他们保留了一些东西，因为他们觉得出于“插入一些奇怪的原因”，那些东西“不适合”放到简历中。因为这种情况很常见，而又如此重要，所以我们把它列出来了。

比如说亚马逊的产品经理杰西卡，曾经应聘亚马逊和其他公司的其他产品经理工作。在简历经过多轮的修改和反馈后，她的简历几乎堪称完美了——只有一个细节例外。她忘了提自己曾经创办过一家游戏公司，雇了几个开发人员和设计师，并督导过一款游戏的开发。这次创业的经历加上她在亚马逊的工作经历，基本上就是她进入任何产品经理面试的金钥匙。

她为什么没把这段经历放到简历中呢？因为某些医疗上的问题，她还没有推出那款游戏。她觉得除非你已经“做成了”，否则就不应该提它。

在类似的情况中，其他人给出了各种各样的理由：“那是为了上课用的”“它不是一个官方的课程项目”“我们还没做完呢”“我们的下载量不多”。

这些理由都不足以把一些东西从你的简历中拿掉。如果对你有帮助，把它列上去。

法则的实现

问问自己：你把什么漏掉了？有没有把你曾经做过的项目（你自己做的，给学校做的，给朋友的公司做的，给编程马拉松做的，等等）漏掉？任何相关的爱好？或者一些有好玩的成就（比如完成了一次铁人三项赛）的兴趣？

简历中放和不放什么没有硬性的和可快速参考的法则。如果某些内容能显得你更有趣或更有吸引力，就把它放进去。

7.3　优质产品经理简历的属性

雇主想找的产品经理应该懂技术，热爱技术，有主动性，有领导力，并且能产生影响。简历为你提供了展示自己背景中这些特点的机会。

然而简历的意义还不止于此。简历本身就是一个产品。它宣示了你的沟通技能、设计技能和你站在用户的角度看问题的能力。

考虑下你的受众：他们关心的是什么，以及你如何展示自己有的那些东西？比如说，如果你

为一家没什么名气的公司工作，你能在简历上简洁地介绍那家公司是什么样的吗？有你能确立信誉的办法吗，比如提一下是谁创办了那家公司？这是你展示“产品设计”技能的机会。

对于很多产品经理岗位来说，展示这些技能很重要。

- **对技术的热情** 如果你有技术技能，或者在科技公司工作过，这可能表明你对技术有足够的热情。如果你没有这些东西，找些其他办法和技术搭上界。你可以通过网络课程学习编码，搭建你自己的网站，甚至外包开发一个项目。
- **主动性** 你可以在大学的俱乐部里，在你创业公司的新生培训计划中，甚至在对技术感兴趣的人的月度聚餐中展示你的主动性。在你的简历上展示这些经验。
- **领导力** 如果你曾经以某种身份管过人，写出来。包括指导/管理一名实习生，或者担任过俱乐部或组织的主席的经历。
- **影响力** 表明你在之前的岗位上有积极的影响。要写清楚你个人曾推动过什么，因为你团队的成就跟你自己的成就相比相关性要小很多。明确指出你曾经构建的、创建的、领导的或实现的东西。避免使用“合作过”和“辅助过”之类没有力量的词语。
- **技术技能** 如果你会编程，在“技术技能”部分列出这些编程语言。这会表明某种程度的熟练性。理想情况下你还要列出特定的项目。
- **注意细节** 这一点更多是要你注意别做什么，而不是做什么。别有拼写或语法错误。要点的尾部统一都用分号（分号是可选的，但要保持一致）。联系信息要正确。

重新审读你的简历，寻找这些属性的迹象。如果缺了某些属性或技能，想办法获取它们，然后加到你的简历上。

7.4 包含什么

你的简历明显应该有你的工作经历和教育背景。那么其他所有的小细节呢？

7.4.1 目标：别写

目标几乎无一例外是浪费空间。我们以一个典型的目标为例来剖析这一点：

“经验丰富的、崇尚行动的技术领导，寻求一个快节奏的、增长性公司的产品管理岗位。”

“崇尚行动”是主观意识，所有人都可以这么说。“经验丰富”这种描述也能从应聘者的简历中看出来。对公司的描述也没什么用。你在应聘这家公司；你肯定对它感兴趣，不管它是不是快节奏的。

目标只是用一种繁琐的方式描述你要应聘的岗位。没必要把已经清楚的东西再写进去。

7.4.2 摘要：少写

摘要很少会有用，基本没例外。如果你的简历足够简洁，它已经是一份摘要了。没必要再用段落的形式重新摘要一次。

另外，大多数摘要都是稀松的，主观的个人描述，比如“动态的”和“行动导向的”。这对读者来说没什么分量。

摘要偶尔能突出下特定的工作成果或岗位职责，以防它们被读者忽略。然而这种情况很少见。恰当的设计几乎总是能让你的亮点显现。

7.4.3 技能：根据需要写

你可能想要在简历上放一个技能部分，特别是在你会编程或者会用设计软件时。忽略那些明显应该会的技能，比如微软的 Word。把你知道如何打开文件，编辑它，然后保存它当作技能列出来。每个人都知道怎么做这些事。

7.4.4 奖励：要写，并且要写得有意义

你应该把自己的获奖情况列上。即便看起来跟技能没有直接关系，但经常能表明你获得的成功，工作努力，或创造性。

很多应聘者会列出他们的获奖情况，但却没挖掘出奖励的意义。他们写的获奖情况就像下面这样：

- 约翰 · 罗伯逊纪念奖（2013 年）。

筛选简历的人不知道这是什么意思。约翰 · 罗伯逊奖是奖励什么的？怎么选的？你因为什么得的这个奖？

理想情况下，你应该在简历中描述这个奖项奖励的是什么，以及它是如何评选的。比如上面这个奖项，可以这样写：

- 约翰 · 罗伯逊纪念奖（2013 年）：在有 75 名学生参加的创业计划大赛中赢得第一名。参赛方案为低成本的太阳能加热泳池。

这样就讲清楚了相关性和评选规则。

7.4.5 活动：有时候写

这要取决于是什么活动，还有你做了什么，活动和兴趣可能会有用。

活动的相关性越强，越应该把它写上。很明显技术活动和创作性工作应该合适。

即便活动本身不是特别相关，你做了什么可能也有关系。比如说，跑步可能和你要应聘的工作没关系（除非你应聘的是健身相关的公司）。然而如果你已经参加了 17 个州的马拉松，并且毕生要参加所有 50 个州的比赛，那么它可以展示你的毅力——并且也会显得你“有意思”。

考虑下你要在简历上呈现的活动。尽量让每一个都有具体的成果。如果你只是另外一个在简历上写着“喜欢爬山”的人，那它一般没什么意义。

7.4.6 项目：写上

项目极其重要。实际上可能是第二重要的东西，紧跟在工作经验之后。

如果你有任何业余项目，一定把它们写到简历上。描述下你做的这个项目是什么，以及成功的指标是什么。

比如说：

- **蛇和梯子（iOS 游戏）** 为多人回合制 iOS 游戏设计 UX，并且雇用外包开发团队实现这个游戏。第一个月的免费下载量达到了一万，并且有 10%转化成了付费用户。

这展现了你承担的领导职责，还量化了你产生的影响。

7.4.7 网站地址：写上

如果你有一个网站或博客，你应该把它的地址放到简历上。

如果你没网站，可以考虑做一个。你的网站应该提供你的简历，以及你的项目的进一步细节（比如截屏）。还可以列出你写过的文章，你发表过的东西，你讲过的课，以及其他有意思的东西。

放一些基本的个人信息也没问题，但网站总体上要专业。不管你是不是把它放上去了，雇主会愿意看的。

7.4.8 社交媒体账号：可能吧

如果你在社交媒体上对技术或与工作相关的问题表现得很活跃，那你的社交媒体账号就可以证明你的价值。但一定要记得把那些显得你很差劲的老帖子清除掉。

7.4.9 学院/大学里的详细情况：看情况

你毕业的时间越长，简历上对学校的介绍应该就越少。什么时候去掉那些条目要取决于是什

么样的活动。这里有一些通用的指导原则。

- ❑ **俱乐部会员和其他“参与”的事项** 只是参加一个俱乐部并不能说明问题。如果不能早点删掉，也可以在毕业后都删掉。
- ❑ **编程项目** 如果你做过编程的项目，可以让它们在你的简历上呆到毕业后 2 到 3 年。如果你能尽早用更有趣的项目换掉它们，那更好。
- ❑ **主要领导岗位** 如果你是俱乐部的主席，或者有作为领导人取得的重要成就，你有理由让它们在你的简历上呆到毕业后 2 到 5 年。只是某个俱乐部的“营销副总”没有太大意义，可以尽快去掉。
- ❑ **创始成就** 成立一家俱乐部，慈善机构，组织一次运动会或其他大型活动，这些都能表明你的主动性，以及你有把事情做成的能力。因此，你有理由让它们在你的简历上停留的时间稍长一些——可能是 5 到 10 年，这取决于你取得的成就如何显赫。
- ❑ **奖励** 这根据奖励情况的不同会有很大变化。一份非常有分量的奖励可以在简历上保留 10 年（甚至更久）。不太重要的奖励，比如在校园编程大赛中赢得第三名，差不多应该在两年以内去掉。

这个指导原则仅仅是经验法则，对你和你的简历来说可能并不适用。对你来说多长时间合适取决它表明了什么，技能有多重要，你是如何展示其他事项的，以及包含那个事项的机会成本是什么。比如说，如果从你的简历上来看，你是一名刻板的极客，并且你应聘的是产品经理，那么你是一名脱口秀小组成员可能应该在你的简历上保留 5 年，甚至更长时间。

一般来说，如果你还在读，那么你的教育经理应该在简历的顶端，但无论如何，工作经验都应该是第一位的。不要把这个当成是一成不变的规则，问问你自己，愿意让那些公司先看你的公司，还是先看你的学校。如果你的工作经历比你的教育背景强好多（或者相反），应该也可以打破这条“规则”。

7.4.10 GPA（平均成绩）

刚出校门的头几年，如果你的 GPA 成绩超过了 3.0/4.0，应该写上。如果你已经毕业 5 年多了，那最少也应该在 3.5 分以上才能留着。理由很简单：除非它真的让你很突出，否则不要把很长时间以前的东西拿出来。

如果你的 GPA 成绩很优秀，那就没有最后的时间点。可以一直放在要点上列出你的专业。大多数人可能不会关心，但它也不占什么地方。所以机会成本是 0。

如果你的大学计算 GPA 时用的不是 4 分制，那么其他人可能很难理解你的 GPA 是什么意思。此时你要试着把 GPA 转换成更有意义的表示方式。比如说，你可以在你的 GPA 后面加上一行（“等同于 3.3/4.0”），或者列出班级排名或百分比。

7.4.11　在线课程和“课外班”

如果你上过在线课程，可以把它们写在简历上。这样可以展示你对学习的热情，以及在那个科目上的专业性，这两个都是好事儿。

如何列出在线课程是有技巧的。如果课程分量足够，你可以把它归在“教育”部分，但放在“更多信息”部分可能更合适。

想办法让这些课程看起来更合理。如果有（好）成绩，也可以写上。或者，如果你已经通过这些课程完成一些有趣的项目，也会很有帮助。

第 8 章 真实的简历：修改之前和之后

即便你的简历已经好到足以成功应聘产品经理，通常也还是有改进的空间。为了让你对可能的改进有更直观的认识，我们决定跟你分享一些成功应聘的真实简历，并给出对其进行改善的例子。

注意：这里给出的“之前”的简历就是应聘者用来应聘产品经理的，不过在某些地方已经做了匿名化处理。简历的匿名化处理包括改变姓名和重要细节，但和实际情况基本一致。比如说，应聘者本来读的是哥伦比亚大学，我们会给改成康奈尔大学。

8.1 理查德·王（匿名）

理查德·王在 2013 年收到了 Dropbox、谷歌和 Uber 的录取通知，用的就是下面的简历。

8.1.1 原版简历

工作经历

ImaginiNow

产品负责人（2013 年 1 月 ~ 2013 年 4 月）

- 带领产品团队拓展队员人际关系；
- 在结识新朋友的请求上达到了 90%的增长率，注册率增长了 97%，病毒式邀请增长了 23%；
- 根据公司愿景、用户反馈和指标推动产品策略，将功能特性一分为二以加强产品的专注性；
- 根据功能特性的想法开发详细规范，并跟设计师合作创建产品原型；
- 用 Mixpanel 定义、实现和分析指标，以便衡量产品的成功程度；
- 确定周冲刺的工作优先级，协调工程和设计团队跟踪进度；

- 为客户培养确立新的渠道，包括可用性焦点小组和用户咨询委员会；
- 应用精益创业方法来设计实验，构建最小可行产品并快速迭代。

Connecto

创始人（2010 年 ~ 2013 年）

- 创建了一个社交网络 app，帮助专业人员互相建立联系，以便发现工作机会和销售线索；
- 筹集到 50 万美元投资，投资者包括贾森·麦克米兰（Vault.com 的初始投资者）和无限创投合伙人；
- 基于客户的需要和竞争格局推动产品愿景、设计和执行；
- 编写功能特性规范，设计用户体验，用 Fireworks、HTML/CSS 和 Axure 创建原型；
- 设计 A/B 测试的着陆页，通过群发邮件、推文以及在 Facebook 上发布内容来增加转化率；
- 定义并分析关键绩效指标以优化用户参与度；
- 用 RoR、HAML、JQuery、CoffeeScript、SCSS、Twitter Bootstrap、PostgreSQL 和 MongoDB 开发网站。

微软

项目经理（2005 年 ~ 2009 年）

- 项目管理团队高级成员，指导 Windows 开发；
- 带领 20%的职能交叉项目为 Windows 开发新技术：编写工程计划，招募十人开发团队，跟踪进度，并向高级工程副总报告结果；
- 协调 30 多个工程团队开拓进展、排列任务的优先级，并实现里程碑目标；
- 在有 860 多人的团队中传达项目更新及目标；为工程高级副总制作状态报告；
- 跟营销合作为微软构建开发者大会开发演示产品；每周跟营销高级副总碰面了解反馈；
- 组织内部技术交流计划来推广新技术；日参加人数达 300 多人；
- 用 C++、C#和 SQL 开发整个组织内使用的项目管理工具；
- 跟工程师一起调试 OS 技术问题，以便完成每日构建。

麦迪逊辅助生活设施

项目负责人（2004 年 ~ 2005 年）

- 带领 15 人团队设计并实现基于 Web 的触屏医疗记录

教育

美国西北大学凯洛格商学院

创新与创业专业工商管理学硕士（2009 年 ~ 2011 年）

加利福尼亚大学圣巴巴拉分校工程学院

计算机科学与工程专业学士学位（2001 年 ~ 2005 年）

其他信息

- 兴趣：探险旅游（印加古道徒步、墨西哥湾航行、阿迪朗达克山脉冰上攀岩）、摄影和以色列格斗术（马伽术）

8.1.2 简历点评

实际上，如果跟其他大部分人的简历比起来，理查德的简历还相当不错，然而它还可以更好。

最重要的是，理查德可以在简历内容中更突出他的成就。他应该对着每份工作问问自己：“我最重要的 5 个工作成果是什么？”然后让这些成果作为简历中的要点。

比如说，看看他最近一个岗位下的要点。有很多模糊的句子，比如“推动产品策略”和“定义工作的优先级”。这些都不是真正的工作成果，它们只是过去的职责。如果你没有展示出自己的影响力，那它真的没多大意义。

另外，理查德在每份工作下都列了很多要点，结果每项要点都显得不太重要了，应该仅限于亮点。

如果他这样是为了让简历看起来很饱满，还需要提供些其他方面的背景信息。也许是在 MBA 期间担任的有趣角色，或者业余时间做过的一些有趣的项目。

最后，他应该考虑在“其他信息”那部分列出他掌握的编程语言。这样做是因为他还有空间（只要把要点精简一下），并且这样会让面试官对他过人的技术背景一目了然。

说了这么多，让我们看看改进后的简历，修改着重解释了他的公司是做什么的，并在要点中更具体地介绍了工作成果。

8.1.3 改进后的新简历

工作经历

ImaginiNow

产品负责人（2013 年 1 月 ~ 2013 年 4 月）

ImaginiNow 是一家由红杉资本支持的社交网站，有 500 多万会员在其帮助下进行社交活动。

- 领导一支由 10 名工程师组成的团队，优化了工程流程，实现了周代码冲刺、敏捷方法和代码审查；
- 在结识新朋友的请求上达到了 90%的增长，注册率增长了 97%，病毒式邀请增长了 23%；
- 领导重新设计网站的项目，开发新的数据驱动方式，实现了 97%的注册增长和 90%的结识新朋友请求的增长；
- 设计了联系 Facebook 和 Twitter 上好友的新功能，在病毒式邀请上实现了 27%的增长；
- 大幅消减了 50%的功能特性清单，加速关键产品开发；
- 设计并领导了有 100 多人参加的 3 次可用性研究。

Connecto

创始人（2010 年 ~ 2013 年）

Connecto 是一个社交网络 app，借助工作计划和销售线索将专家连接在一起。会员数最多时达到了 100,000 余人，月页面访问量实现了千万以上。

- 成功领导公司融资工作，筹集了 50 万美元的投资，投资者包括贾森·麦克米兰（Vault.com 的初始投资者）和无限创投合伙人；
- 构建了最初的原型/最小可行性产品，用户可以根据“需要”快速搜索他们的 LinkedIn 连接；
- 领导 4 人团队（两名工程师、一名设计师和一名测试员）通过了两次转折，重新定义了战略，设置并领导团队实现新的愿景；
- 开发了公司仪表板，可以追踪用户预订情况；
- 跟两名工程师一起用 RoR、HAML、JQuery、CoffeeScript、SCSS、Twitter Bootstrap、PostgreSQL 和 MongoDB 开发网站。

微软

项目经理（2005 年 ~ 2009 年）

- 项目管理团队高级成员，指导 Windows 的开发；
- 带领 10 ~ 20 名员工组成的跨职能团队为 Windows 开发新技术：解决问题以完成每日构建，管理项目日程，协调项目目标；
- 组织内部技术交流计划来推广新技术，日参与人数达 300 多人；
- 从 4 个团队的 30 个工程师那里汇集信息，为工程高级副总制作周状态报告；
- 用 C++、C#和 SQL 开发新的项目管理工具，并推广到 25 个项目经理的团队中使用；
- 为微软构建开发者大会开发演示产品（参会人员有 5000 多名）。

麦迪逊辅助生活设施

项目负责人（2004 年 ~ 2005 年）

- 带领 15 人团队设计并实现基于 Web 的触屏医疗记录。

教育

美国西北大学凯洛格商学院

创新与创业专业工商管理硕士（2009 年 ~ 2011 年）

- **啤酒迷**（主席/创建者）：成立啤酒俱乐部，并把该俱乐部发展到 100 多名会员，从当地啤酒厂筹集 2 万美元的赞助；
- **启动加速器**：接受凯洛格启动加速器，为创业公司准备的孵化器录取率为 15%。

加利福尼亚大学圣巴巴拉分校工程学院

计算机科学与工程专业学士学位（2001 年 ~ 2005 年）

其他信息

- 编程语言：RoR、C++、C#和 SQL；
- 兴趣：探险旅游（印加古道徒步、墨西哥湾航行、阿迪朗达克山脉冰上攀岩）、摄影和以色列格斗术（马伽术）

8.2 保罗 · 昂特伯格

保罗一路打拼从技术支持跻身顶级创业公司的产品经理，下面是他最新的简历。

8.2.1 原版简历

软件产品经理

专业领域：最低可行性产品、市场分析（产品/市场匹配）、敏捷开发、线框图/原型，在创业环境中的 10 倍生产力。

工作经验汇总

加利福尼亚州红木城 Pricelock 有限公司（2010 年至今）

高级产品经理

高盛和 Artiman 投资的创业公司。有多个基于 Web 的金融服务、风险管理和能源交易产品，负责设计和新产品规范说明，以及已有产品的提升改进。

- 为能源交易商推出在线市场，进行价值 20 多亿美元的能源产品交易；
- 以产品所有人的角色提高开发团队生产力；
- 构思新的服务和产品，帮助公司从新的投资者那里吸引投资。

加利福尼亚州旧金山市 TechExcel 公司（2003 年 ~ 2010 年）

副总监，产品管理（2006 年 ~ 2010 年）

产品经理（2004 年 ~ 2006 年）

管理产品组织。为 TechExcel 的主力产品线制定产品策略，跟客户一起定义需求和功能特性，指导外包和分散在全球的开发团队。面向潜在客户、消费者、分析师和内部团队提供技术解决方案。

- 引入 Scrum 敏捷方法，明显改善产品开发过程；
- 创建系统根据经验来确定优先级，平衡客户和 TechExcel 的商业需求；
- 把新产品带入市场，并让软件的收益翻了一番。

高级解决方案工程师（2003 年 ~ 2004 年）

为业内领先客户提供技术咨询、培训和持续支持。

- 为 ALM 和 CRM 实践及软件部署提供咨询；每次部署都得到了客户最高程度的认可（客户评分为满分 5 分）。

加利福尼亚州旧金山市旧金山联合校区，微软顾问（1999 年 ~ 2003 年）

高级数据库架构师

- 管理多平台网络环境，监督一个 DBA 团队为旧金山市所有学校的 100 多个数据库系统提供支持

教育背景

旧金山州立大学计算机科学学士

编程语言及工具

PHP、SQL、Visual Basic、C、JavaScript、HTML、CSS、Excel、Balsamiq、PowerPoint、Apache、MySQL、Linux、AWS、Oracle、Marketo、Google Analytics、Wordpress、Photoshop、Fireworks、SalesForce

8.2.2　简历点评

保罗简历中的主要问题是不够具体。他会提到一些成就，比如“将新产品推入了市场”，但没能讲清楚是哪些项目；同样，他说自己“提升了开发团队的生产力”，但也没有解释自己究竟是如何实现的。

还有一个相对来说比较小的问题，但也应该予以纠正：格式。他把职位头衔都弄成了黑体，就好像在说："嗨！瞧瞧！我是一名产品经理！"这个细节并没有什么意思，很多应聘者都是产品经理，实际上他应该重点突出他所在的公司。

"专业领域"那部分内容对提升简历没什么用，最好能通过工作成果展示这些技能。

一些工作之外的项目或活动也会有所帮助。如果他有，应该列出来。

下面是一份改进过的简历。

8.2.3 改进后的新简历

工作经历

Pricelock 公司（加州红木城，2010 年至今）

高级产品经理

高盛和 Artiman 投资的创业公司。旗下拥有有多个基于 Web 的金融服务、风险管理和能源交易产品。负责设计和新产品规范说明，以及已有产品的提升改进。

- 为能源交易商推出在线市场，进行价值 20 多亿美元的能源产品交易；
- 设想并推出了一个概念验证的预测交易功能，经证明能够增加 25%的利润（吸引到了一笔一千万美元投资）；
- 为提高团队生产力和产品质量，确立并推出了更好的开发实践，包括易读性指南、时间表估计、敏捷开发、"不开会的星期三"，以及定期 bug 会审。

TechExcel 公司（加州旧金山，2003 年 ~ 2010 年）

副总监，产品管理（2006 年 ~ 2010 年）

产品经理（2004 年 ~ 2006 年）

管理产品组织。为 TechExcel 主力产品线制定产品策略，跟客户一起定义需求和功能特性，指导外包和分散在全球的开发团队。向潜在客户、消费者、分析师和内部团队传播技术解决方案。

- 主导并推出数据分析工具，让医生可以更精确地跟踪临床试验数据，从而减少了 95%的错误；
- 通过分析使用情况指标实现软件续费收益翻一番，并为产品续费设计了新流程；
- 引入 Scrum 敏捷方法并带领 5 个工程师做每日例会；
- 为医疗记录产品设计针对政府组织的商业计划，实现 30%的效益收益；
- 为 FDA 批准产品设计框线图并构建原型，担任新事业部门高管。

8

高级解决方案工程师（2003 年 ~ 2004 年）

为业内领先客户提供技术咨询，培训和持续支持。

- 为 ALM 和 CRM 实践及软件部署提供咨询；每次部署都得到了客户最高程度的认可（客户评分为满分 5 分）

旧金山联合校区微软顾问（加州旧金山，1999 年 ~ 2003 年）

高级数据库架构师

- 管理多平台网络环境，监督一个 DBA 团队为三藩市所有学校的 100 多个数据库系统提供支持

教育背景

旧金山州立大学

计算机科学专业学士

项目和活动

- **Commercy（2012 年）**：为本地的自行车商店构建了一个定制的电子商务网站，支持运费计算，在线商店管理和定制主题（PHP、CSS、HTML）；
- **SF 辅导，**主席（2010 年 ~ 2012 年）：负责一个涵盖 100 个低收入家庭和 40 个导师的辅导计划。设计了新的导师筛选计划，年度周转同比从 50%降到了 20%。

语言及工具

- PHP、Visual Basic、C、JavaScript、SQL/MySQL、HTML、CSS
- Balsamiq、Apache、AWS、Oracle、Marketo、Google Analytics、Photoshop、Fireworks、SalesForce

8.3 阿米特 · 阿加瓦尔（匿名）

阿米特 · 阿加瓦尔被谷歌录用为助理产品经理实习生。

8.3.1 原版简历

教育背景

斯坦福大学

学士，预计 2013 年 5 月毕业；GPA：3.65，主修专业：计算机科学；Facebook 夏季黑客大赛 2011 年获胜者

MellowBrooks 宪章高中

2009 年 7 月毕业，GPA：4.0/4.0；SAT：800 R，800 M

奖励/荣誉：毕业生代表、全国优秀入围奖、全国 AP 学者、麦卡利斯特奖学金、亚历克西斯 X·永利奖学金

工作经验

Codesion

软件工程师实习生，2011 年夏

构建了一个将 bug 数据库与目标系统持续同步的 Python 系统，目标系统包括搜索引擎、集成平台等，同时处理故障状况。

斯坦福大学法学院

研究助理，2010 年春季 ~ 2010 年夏季

给吉拉德·撒们博士做助理。分析来自密歇根 HRS 的数据，以便确定退休对健康的影响。研究发表公司管理方面的文章。

斯坦福大学商学院

研究助理，2010 年夏季

跟博士生迈克尔·詹姆森和帕特里克·Revik 博士合作一起收集所有上市公司的信息。致力于构建完备的行业分类标准。

活动

- ❑ 斯坦福新贵，主席（2010 年 2 月至今）
- ❑ 斯坦福大学的 ACM 小组成员。在 ACM 区域赛中排名第 5（2010 年 9 月至今）
- ❑ 斯坦福大学俱乐部足球队守门员（2009 年 1 月至今）

斯坦福大学无伴奏合唱队，2010 年 1 月至今

技能

计算机编程语言：C、Python、Java、SQL、JavaScript（熟悉）、Ruby（熟悉）、Scheme（熟悉）

8.3.2　简历点评

阿米特的背景实际上相当棒，但遗憾的是这份简历让他的经验显得平淡无奇。简单地说，他抓错了重点。

像其他很多（或大部分）应聘者一样，他过于强调职责，忽视了自己取得的成果。他的简历应该用 3 到 5 条要点列出他对一个组织的影响。一定要突出重点！

还有一点比较有意思，他的简历中没有提到项目。他是学生，所以几乎可以肯定他一定做过一些项目，哪怕不是出于兴趣，也至少帮学校做过。一定要把这些项目列上。

再看他的活动，混杂着各种各样的事情。其中有两项活动对我们来说没什么意思；他不太可能靠唱歌和踢球这种事得到一份工作。有一个我们很感兴趣，但它混在了其他事情中间。“新贵”活动可能是我们感兴趣的，但因为没有详细描述，读者也不可能知道究竟是什么情况。

我们会根据这些情况重新修改他的简历。删除一些不重要的条目（高中、运动等），以便突出那些真正吸引人的成果。

8.3.3　改进后的新简历

教育背景

斯坦福大学

计算机科学学士（2009 年 ~ 2013 年）；GPA：3.65

工作经验

Codesion

软件工程师实习生（2011 年夏）

- 实现了一个同步工具，将新 bug 数据库与主 bug 数据库同步，以减少新用户切换的成本（Python）。
- 用搜索 API 扩展工具，以便同步在线论坛上提交的 bug 报告，并解析报告（Python，谷歌搜索 API）。
- 基于阿历克斯·陈的权重分析算法设计新的算法，用来从 bug 报告中提取元数据（操作系统，里程碑等），并将算法优化了 85%。

斯坦福大学法学院

研究助理（2010 年春季 ~ 2010 年夏季）

- 提议并构建系统来自动分析来自密歇根 HRS 的数据，以便确定退休对健康的影响。
- 在高级机器学习（AML）杂志上跟吉拉德 · 撒们博士共同署名发表论文（2010 年夏），吉拉德 · 撒们博士在 AML 的年度大会上展示了这项研究成果。

斯坦福大学商学院

研究助理，2010 年夏季

- 收集所有上市公司信息，并帮忙构建完备的行业分类标准。
- 编写 Python 脚本从网上信息源中提取数据，节省 98%人力及时间成本。

项目

- **EdU Projecto（iPhone App，独立制作）：**项目管理 app，帮 CS 学生使用学校的项目管理时间线和工作流。iOS app 在线商店头三个月下载量达到了 1000 多次，评分为 4.7/5.0。
- **广告牌处理器（C++，课堂项目）：**从广告牌的照片上提取文字的程序。准确率高达 95%，平均准确率为 85%。

奖励和活动

- **Facebook 夏季黑客大赛（2011 年）：**在一个周末黑客马拉松大赛中从 47 个团队中脱颖而出，取得了第一名。
- **ACM 区域赛（2010 年）：**关于数据结构和算法的竞赛，在 150 多个团队中取得了第 5 名的名次。
- **斯坦福新贵，**主席（自 2010 年 2 月至今）：组织半年度的黑客马拉松，有 100 多名学生参与。从 30 多家赞助商筹集到 1 万美元奖金，其中包括谷歌、微软和 Facebook。

附加信息

- **编程语言：**C、Python、Java、SQL、JavaScript（熟悉）、Ruby（熟悉）、Scheme（熟悉）
- **其他活动：**斯坦福大学俱乐部足球队守门员（2009 年 1 月 ~ 2013 年 5 月）；斯坦福大学无伴奏合唱队（2010 年 1 月 ~ 2013 年 5 月）。

8.4 亚当 · 凯兹维尔

亚当凭这份简历收到了 GigaOM 的录取通知。

8.4.1 原版简历

总结

早期采纳消费者价值识别才能的人。擅长创建产品愿景，分析业务绩效。擅长判断消费网络

行业的趋势和机会，热衷于构建受到百万用户喜爱的产品。

AOL

AIM.com 和 Games.com 产品经理（2011 年 7 月 ~ 2012 年 3 月）

加入贾森·谢伦的团队，帮助重新推出两个关键特性。就 AIM 而言，重点是 Web 体验，并帮助设计着陆页和新的用户流程，还负责处理跨平台接入的用户支持。

- 整合 AIM 新老用户的欢迎体验，向他们介绍经过完全更新的产品。分开来说，用 UserVoice 收集、整合及响应接入的反馈——确定可解决的最高优先级问题，并将汇总结果分享给团队；同时，开始创建细化过的指标仪表盘。
- 在加入 AIM 团队之前，帮 Games.com 确定刷新策略。重点是面向平板电脑的设计，向 AOL 消费者产品组的高级领导人汇报工作。

Hotwire

产品经理（2009 年 2 月 ~ 2011 年 7 月）

将新的功能特性转化为详细业务需求，并在整个组织内沟通它们的设计和功能。跨越多个垂直行业，设置完整的项目及临时工作，主要使用瀑布式 SDLC。

- 跟一名内部设计师和第三方团队一起合作，创建 Hotwire 第一版专为移动端优化的网站。主导决定网站应该包括/去掉哪些部分以及内容的呈现方式。
- Hotwire 国际网站启动团队成员，这是 2011 年的关键性战略项目。帮助快速搭建了一个模块化平台，在提供核心功能的同时又易于迭代。
- 作为节约成本练习的一部分，为交付一个自动化、可扩展的常规数据采集任务解决方案设计了模板和处理流程，这个解决方案用到了亚马逊土耳其机器人。

LiveJournal

产品经理（2008 年 4 月 ~ 2009 年 1 月）

从想法到推出，负责全面的产品生命周期管理。编写概念文档和 PRD，创建初始模型，在产品推出前后收集指标。同时跟美国和俄罗斯的开发人员合作，交付对预订增长有帮助的关键项目。

- 重新设计账户层级管理，扭转付费预订用户数下滑趋势——让用户更容易理解他们在各个账户层级上会得到哪些功能特性。
- 用流量指标确定唯一产品定位机会，用见解吸引更多广告商。研究竞争格局和市场机会，同时用 PPT 向高级管理层汇报对产品趋势的见解。

雅虎

业务分析师（2004 年 11 月 ~ 2008 年 2 月）

以天、周、月为日期单位，向高管团队报告并分析个人产品总体表现。跟产品团队密切合作定义并跟踪产品发布的影响。

- 创建并维护日和周仪表盘，跟踪季节性趋势，100 多个产品每周及每年的表现及推荐指标。
- 把产品关键驱动因素的知识跟个人网站用户体验知识结合起来，向产品团队呈现的见解可以促成低成本，高影响的功能特性开发。
- 协调产品团队定义、跟踪、分析每次发布的好处。决定发布最有可能影响哪些指标，并计算基线来确认它们的表现是否处于期望的范围内。

TXU 能源

市场分析师（2001 年 12 月 ~ 2004 年 9 月）

首先做技术写手，后来转做市场和商业信息小组的分析师。负责收集汇报需求并将其变成技术规范。

- 主持有多个主管参加的需求收集会议，推动标准化指标的创建及可重复的营销绩效报告，它们要在多个业务部门中使用。
- 用 SQL、Crystal Reports 和 Access 提取数据，然后通过解析和合并域在 Excel 中清理并格式化，在透视表和图表中汇总。创建宏来加快所有可重复的流程。

教育背景

马凯特大学，1997 年 8 月 ~ 2001 年 6 月（威斯康星州密尔沃基市）

理学学士，主修工业工程

其他

在整个消费网络上都很活跃。最常出现在 Twitter（@KAZ）、Quora 和 Instagram 上。

8.4.2　简历点评

亚当的简历确实只需要一页纸，但它还是很长。文本段落设置逻辑模糊，容易让招聘人员忽视其亮点。

此外，他的简历也不够具体和清晰。

比如说，看看他在 AOL 的工作经历。有一些东西是关于“欢迎体验”的，但没讲清楚究竟

是什么。之后，他说自己用 UserVoice 收集反馈，分享结果，开始做些事情，并帮着做些其他的事情。这些全都不是真正清晰的成果。

他描述自己在 HotWire 的工作时也有类似的问题。他跟一些人“合作”，“帮着”做一些其他事情，是另一个小组的“成员”，参与了一些“练习”。这些都缺乏它们应有的“吸引力”。

写到 Live Journal 时，我们看到了一些清晰的成果，但仍然有待提升。“逆转了付费预定用户数下降趋势”部分很棒，但如果能有数据支撑就更好了。他说自己用流量指标确定定位，吸引广告商。“吸引广告商”陷在了要点的中间，并且没有数据支撑；即便读者注意到了，也不会觉得特别可信。

亚当的简历绝不能算糟糕；实际上比大多数简历都强，然而它还可以更好。

亚当接受了这个建议并重新编写（并更新）了他的简历。以下是部分修改内容。观察一下怎么样。

- 在 AOL 的工作更清晰了（现在我们能够弄懂欢迎体验是什么了），并且更侧重于他自己的工作成果。
- 在 HotWire 的工作展示了更多的领导力。现在是他领导了什么，而不是跟谁合作。还有数据支撑他是怎么降低成本的。现在我们还了解到它不只是一项“练习”，而是做了一些有意义，有影响力的事情。
- 就 Live Journal 而言，提供了预订用户数变化趋势的数据，提升了可信度，在快速浏览时关注度更高。同时，还明确指出自己吸引了两个新的广告商（如果他能说出带来多少收入可能会更好）。另外，还增加了一个关于连接增长的要点。
- 总结部分更有意义。所有人都可以宣称自己有“识别消费者价值的天赋”，所以那没太大意义。新的总结让人感觉更实在。
- 保留了自己的社交网络用户名，这是让公司更加了解你的好办法。

他的要点还是有点长，但他的简历已经有了大幅的提升。之前我们看完他的简历后只会留下他负责什么之类的大致印象。现在我们不仅知道他负责一些有趣的事情，还知道他取得了什么成果。这是你要在自己的简历中尽力去做的事情。

8.4.3 改进后的新简历

有 5 年多经验的消费网络产品经理。工程背景，换岗到产品经理之前做业务分析师。专注于发现问题，汇集最好的想法，并在此基础之上迭代构建优秀的产品。

工作经验

GigaOM

产品经理

负责 Gigaom.com 基于 WordPress 博客的主要内容，包括首页和故事页。带领交互式网站的重新设计。报告并分析通过谷歌 Analytics 和 Chartbeat 得到的关键产品指标。通过 Asana 和 Github 驱动敏捷开发。

- 带领 GigaOM.com 产品的响应式重新设计。设定总体方向以及 Gigaom 最新最大项目。推出之后，移动端流量在线时间翻倍，非峰值时间访客流量增长 20%。
- 接手一个已经停滞数月的“分析师联线”产品，并在 PPT 中创建了一个可点击的模型。促成了团队内迭代的改善，并在一个月后交付了一个经过打磨的产品。
- 改写 WordPress 帖子格式，交付了一个定制化帖子类型。让作者的产出增长了 25%，并引导记录单一访问者和浏览量水平。

AOL（AIM.com 及 Games.com）

产品经理（2011 年 7 月 ~ 2012 年 3 月）

加入贾森·谢伦的团队，帮助重新推出两个关键特性。就 AIM 而言，重点是 Web 体验，并帮助设计到达页和新的用户流程，还负责处理跨平台的接入用户支持。

- 为 AIM 新老用户勾勒并设计了欢迎体验，设计登录页面及特性演示功能。
- 管理 User Voice 账号，收集、整合、响应上百万用户发来的反馈；通过分析状态确定可解决的最高优先级问题，并带领团队解决这些问题。
- 带领 AIM 团队创建并沟通 Games.com 的刷新策略；包括取得关键管理人员的支持，并重新制定路线图、确定新目标。

Hotwire

高级功能设计师（2009 年 2 月 ~ 2011 年 7 月）

将粗略功能特性请求转化为详细业务需求。确定功能特性的设计及职能规范，并在整个组织内沟通最终规划的状态。跨越多个产品垂直行业工作，完成排上日程的项目及临时工作，主要是用瀑布式 SDLC。

- 指导一名内部设计师和第三方团队开发 Hotwire 第一版移动端优化的网站；主导决定网站应该包括/去掉哪些部分，以及流程应该如何优化。
- Hotwire 国际网站启动团队成员，制定计划搭建一个模块化的平台以加快迭代的速度，以迅速适应新市场。
- 降低 90%常规数据收集任务成本，使用亚马逊土耳其机器人为一个自动化的、可扩展的解决方案创建模板和处理流程。

LiveJournal

产品经理（2008 年 4 月 ~ 2009 年 1 月）

负责全面管理产品生命周期，编写概念文档和 PRD，创建初始模型，在产品推出前后收集指标。同时跟美国和俄罗斯的开发人员合作，交付对预订增长有帮助的关键项目。

- 重新设计账户层级管理，让用户更容易理解在不同账户层级上会得到哪些功能特性，扭转一年期付费预订用户数每个月下滑 3%的趋势，实现了每个月 2%的增长。
- 导航复杂的隐私设置，设计了一个更新“找朋友”功能，在刚开始的 90 天内就让新用户连接数增长了 10%。
- 分析研究流量指标，并推出了一些期刊，吸引到了两个新的广告客户。

雅虎

业务分析师（2004 年 11 月 ~ 2008 年 2 月）

按天、周、月和季度向高管团队报告并分析个人产品的总体表现。跟产品团队密切合作定义并跟踪产品发布的影响。

- 创建并维护日和周仪表盘，跟踪季节性趋势，跟进 100 多个产品每周、每年表现及推荐指标。
- 综合产品关键驱动因素和个人网站用户体验信息，促成产品团队低成本，高影响的功能特性开发。
- 协调产品团队定义、跟踪、分析每次发布的益处；决定发布最有可能影响哪些指标，并计算基线来确认它们的表现是否处于期望的范围内。
- 向产品团队呈现关键产品驱动因素的见解，促成低成本、高影响的功能特性的开发。

教育背景

马凯特大学（威斯康星州密尔沃基市）

工业工程科学专业学士

其他

- 在整个消费网络上都很活跃；
- 特别是在 Twitter（@ KAZ）、Quora 和 Instagram（@kaz）上；
- 更多信息：about.me/kaz。

第 9 章 求职信

求职信有利有弊。有些公司非常重视，有些公司则根本置之不理。很多公司会要求，但也不是特别重视。

一封好的求职信能把你的背景和工作要求连起来。如果你的背景不是“很理想”，这就变得特别重要。你可以用自己的办法告诉读者，尽管乍一看可能不是，但你确实还不错。

9.1 一封优秀产品经理求职信的要素

大多数求职信都很平庸，只是用段落的形式重复了一下应聘者的简历。这可能会展示一下基本写作技巧，但不会有太大影响（好或坏）。

一封优秀的求职信应该具备以下特点。

- **短** 求职信应该保持在 200 ~ 250 个词左右。冗长的简历很可能不会有人去看，并且很有可能会包含很多不必要的细节。此外，太啰嗦也会显得你很差。
- **显现激情** 一个理想的产品经理应该对技术和行业充满激情，这种激情应该在求职信里体现出来，让 HR 明白你为什么对这份工作感兴趣。
- **展示技能** 看看产品经理应该掌握的那些技能——通用的和针对某一岗位的——思考你的背景与其如何匹配或不符。特别要注意那些你已经掌握，但在简历中不好体现的技能；用求职信展示这些技能。
- **契合公司的文化** 公司是欣赏“商务人士”，还是更喜欢一些更有天赋的人？在你的求职信中要有对应的表现。一家有趣的、离奇的公司可能更喜欢有趣的、离奇的求职信——特别是在你的简历看起来就像一个呆板的商业人士时。
- **精心编写** 你的求职信是一篇写作范文，也应该被如此对待。也就是说绝不能有编写或语法错误，还应该简洁清晰，而不是华丽的、描述性的散文。一定要注意你的句子，确保它们不会太长或太复杂，应保持句子长短的多样性。

最终，你的求职信是让你的背景跟“完美的”产品经理相匹配的办法，这也是求职信存在的价值。

9.2　求职信模板

一封传统的求职信应该像一个相当标准的模板。段落（准确地说是第 2 和第 3 段）的顺序可以重新调整，但一般都应该包含以下要素。

9.2.1　确定读者

一句简单的“敬爱的____”就行。如果你知道读者的姓名，填上去。除非你确实知道读者是男性，否则绝对不要说“敬爱的先生”，你很可能因此被拒掉。

> 敬爱的____，

9.2.2　开篇

在第一段里，简要地介绍一下你是谁，以及你在找什么样的岗位。没必要特别指出你的姓名，有签名呢。如果你知道公司里与这个职位有关的人，或者有趣的方式，在这里提一下就很好。

> 我最近参加了 WYSIWYG 开发者大会，当 ChattyCha 的 CEO 提到 API 团队有一个产品经理的工作机会时，我觉得兴奋极了。作为一个从开发转型过来的产品经理，我相信这个岗位对我的背景和兴趣来说会非常合适。

9.2.3　第二段

在这一部分，阐述自己的背景跟这个岗位怎么匹配。这里不应该是你工作角色的汇总，那些都应该放在简历里，这一段要做的是把你的技能及成就跟公司所寻求的东西连起来。

这段应该突出你的软性技能，并以工作成果作为支撑。

> 我是一个痴迷于技术，并喜欢激励他人实现困难目标的人。我在微软做开发的第一年，采用一种新技术解决了一个关键框架中的顽疾。我招募并训练了一个由三名工程师组成的小团队，并在后来成为了相关功能特性的开发领导。我将这种热衷于解决问题的热情带到了 Waffle，一个专注于企业安全的创业公司，并在那里成功推出了公司的主力产品——当时很多人认为它过于复杂，肯定会失败。

9.2.4　第三段

接下来要解释为什么对这个岗位有兴趣。这一段不要太长，因为前面那一段更重要。你只要

展示出自己对这个岗位的重视程度，表明它不只是你应聘的又一家公司。

> 我对 ChattyCha 的使命感到兴奋，我喜欢有深度的技术挑战。尤其是对开发者工具这一领域，我特别感兴趣，因为我在做个人项目和专业编码项目时都体会过那种因工具不足而带来的挫折感。我迫切地希望能贡献自己的力量来改善这一领域。

9.2.5　第四段

最后一段很短，只是用“谢谢您”作为信件的结尾。

> 感谢您的阅读，期待得到您的答复。
> 真诚的，
> （你的名字）

9.3　一封优秀的求职信

按照上面的模板写求职信不能帮你赢得什么大奖，但它基本上能解决你的问题——让 HR 考虑你的简历（或者至少是看一眼）。

极少数情况下，一封求职信可以大大提高应聘者的胜算。

看看以下这封求职信，应聘者想要加入一家教育类创业公司：

> 可能您已经注意到了，我的 GPA 只有 2.7 分，学术真不是我擅长的领域。但我喜欢学习，正是学术界和学习之间的差异引起我对 FusionEd 的强烈兴趣。
>
> 作为学生，我表现平平。大多数学校都以成绩为主，结果就是学生在自己不喜欢的科目上也得花费很多时间。还有，如果我想更深入地探索某个主题，也缺乏相应的资源和结构，而 FusionEd 那种儿童鼓励教育的方式会解决这个问题，所以我想成为其中的一份子。
>
> 把学术放一边，我真的喜欢学习，特别是喜欢学习技术。我去年自学了编程，还参加了几次黑客马拉松。最近一次黑客马拉松，我是唯一获奖的个人。我的 GitHub 账号上有我做过的一些项目，其中还有一个是针对学生的。
>
> 我在 Colapa 做产品经理时，领导一个有 6 名开发人员的团队。我的日常工作包罗万象，从功能特性设计到市场调研，甚至还有简单的编码。我身兼数职，这也是我喜欢这个工作的原因之一。我最自豪的是找到一种办法重新定位我们的产品，进入了企业市场，目前已经有 15%的收益。
>
> 不久之前我曾致力于为 Colapa 开发一个 API。我知道 FusionED 在朝着类似的方向发展，我很乐意再次迎接这样一个挑战。我坚信自己对技术、学习和领导力的热爱，加上我对教育现状的不满，会促使我对 FusionEd 产生影响。

这封自荐信并不一定适用于所有公司，但放到这家公司来看确实是非常优秀的。它很真诚，只是开头的地方有点怪异，它的语调对于一家以改变教育行业为使命的小型创业公司来说非常合适。

在技能上，这位应聘者展示了：

- 主动性（学习编程）
- 对技术的热情（学习编程和参加黑客马拉松大赛）
- 愿意承担风险（参加黑客马拉松大赛，尽管资格上有很大差距）
- 智力（赢得了黑客马拉松大赛）
- 领导力（重新定位现有产品）
- 喜欢创业（加入创业公司以及“身兼多职”）
- 成功（成功地重新定位了产品）

应该用这种方式剖析你的求职信，认真考虑每句话的引申含义。你的求职信展示了哪些技能和特点？记住，不用在求职信中把你所有的经验都列出来，那是简历的内容。

第10章 研究公司

如果你了解一家公司的来龙去脉，那么在面试时就能打动你的面试官。你可以问更多有趣的问题，给出更有见地的答案，以此展示对职位的更多热情。

不过，面试官有时也会忘记，并不是公司之外的所有人都能像他们那样充分了解公司的产品。他们可能会因为你没注意到一些看起来“明显的”细节而粗暴地对你作出评判，你自身也可能因为缺乏某些背景知识而回答不好某些问题。

下面大体谈谈在面试之前，你该了解公司的哪些具体情况。

10.1 产品

首先，应该深入了解公司的产品，大概知道“它是做家庭自动化工具的”这种了解程度是不够的。了解下面这些信息能让你脱颖而出。

- **产品** 公司开发的产品或特性序列，以及各产品间的配合方式。
- **竞争对手** 具体有哪些竞争对手，以及公司与这些对手的差异化特点。
- **客户/市场** 公司的目标市场是什么？现在有二级市场吗？或者你有建议公司要进入的市场吗？
- **收入** 公司怎么盈利？你如何建议公司赚取更多的利润？如果它不赚钱，你想探索什么样的营收策略？
- **客户满意度** 客户对产品的感觉如何？他们喜欢或不喜欢什么？最常见的抱怨和问题都有哪些？
- **指标** 如果可能，尽量了解一下公司的关键指标。找到准确的数字可能有困难，但至少能明白他们在哪些指标上表现不错，哪些是他们正在努力去做的。它有多少用户？转化率如何？增长率怎么样？
- **新闻和传言** 有关于公司的有趣新闻报道吗？有传言公司要做什么吗？不要只是看看。要形成自己的观点。

想要了解这些信息，可以查看公司网站上“关于我们”的页面，以及他们的博客、招聘网站、证监会档案、报刊文章、博客文章、支持页面，以及其他能在网上搜索到的任何相关信息。

同时，还应该亲自使用产品，并且以多种用户类型广泛使用。如果是有免费用户和付费用户的产品，尽量两种情况都试一下（如果可能的话）。找出喜欢与讨厌的地方，要特别关注那些让人不尽满意的地方。

10.2　战略

你不仅应该知道公司在做什么，还要知道为什么这么做。知道“为什么”能帮你按照公司的世界观回答问题。比如说，如果公司想要“改变世界”，那你就应该在自己的答案里加入那种热情。或者如果公司要积极地向全球扩张，你可以讨论全球扩张和国际化事务。

知道“为什么”意味着了解下面这些内容。

- **使命**　了解一下公司的使命（或宗旨），以及它是如何实现这个使命的。尽量具体一些。
- **战略**　你觉得公司的战略是什么？有没有失误？
- **优势**　产品的卖点是什么？公司是如何利用那些卖点的？究竟是什么因素让公司或其产品取得了成功？
- **劣势**　公司和产品面临的主要问题是什么？如何处理那些不足，或者他们只是接受了？
- **挑战**　公司现在面临的最大挑战是什么？你怎么看待他们对那些挑战的应对方式？他们都克服了哪些困难？
- **机遇**　未来（技术或行业内部）有能为公司创造机会的东西吗？
- **威胁**　同样的，未来有没有可能会威胁到公司成功的东西？
- **未来**　你觉得公司未来掌握着什么？想想能自然融合的新产品或功能特性。

这些主题。你了解的内容可能不止上面这些，那很好。如果你能给出一个令人信服的理由，指出产品为什么应该（或不能）成功，并能预测出反方的理由，那么你准备的还比较充分。

10.3　企业文化

有些应聘者过于关注产品了，以至于他们忘了公司是由人组成的（然后再由他们制作产品）；或者他们会在网上快速地搜索一下面试官的相关资料，但并没有考虑他们真正关心的问题。

- **企业文化**　公司的企业文化是什么？公司的招聘页面可能会提到一些，但那明显会偏向他们想要表现出来的形象。在网上找找应聘者、现役员工和前任员工的评价。
- **价值观**　公司的价值观是什么？我们说的“价值观”是指一切对他们来说重要的东西，显式的或隐含的。要了解这个，可以看看采访创始人的访谈录，想想他们的企业文化和

产品。“价值观”也可能包括“快速行动”（Facebook）或者“不作恶”（谷歌）之类的东西。

- **发展历程** 公司做了多长时间了？是怎么开始的？他们是否仍坚持初衷？还是已经改变方向了？
- **面试官** 如果你知道谁会面试你，可以在网上搜搜他们的信息。但也别搞得太吓人，恭贺面试官新婚之喜就显得有点儿“跟踪狂”了；最好不要涉及个人隐私问题。
- **关键人物** 公司是谁创办的？他们之前是做什么的？如果是创业公司，投资者是谁？公司里有知名的人物吗？不仅要了解这些“事实”，还应该想一想创始人和其他关键人物的背景是如何影响公司的？
- **组织** 公司有多大？公司是怎么组织的？是不是所有人基本都是向 CEO 汇报工作，或者有严格的等级划分？

了解这些不是为了打动面试官（可能也不会），而是用这些信息建立起对一个公司的初步印象，同时也能判断公司是否能适合你。有时应聘者过于注重得到工作，以至于忘了考虑那份工作是不是真的适合他们。

这些信息还有一个用处：它能帮你确定对行为问题的反映，比如你如何影响队友或如何实现一个决策。

如果可以，尽量把这些知识拼到一起，也就是将企业文化跟战略、以及公司的其他部分相匹配。比如说，亚马逊的名声是它受金钱和指标驱动，还非常节俭。是因为亚马逊所做的业务利润很低吗？是因为亚马逊很长时间没有盈利了吗？这会如何影响它的战略？

10.4 角色

最后，要知道自己应该如何融入公司。这意味着要注意以下事项。

- **角色** 你应该了解产品经理在这家公司的角色。如何定位技术？如何作出决策？
- **创意产生** 创意是从哪里来的？一些公司是“自下向上”，创意来自于开发人员和产品经理，然后他们用自己的愿景说服高管；一些公司是“自上向下”，高管提供公司的长期愿景，产品经理奉命实现这一愿景。
- **切合实际与疯狂** 虽然相对于疯狂的怪主意而言，多数公司都会稍微倾向于切合实际的想法，但每家公司都会坚持维持两者之间的平衡。要了解公司是更喜欢大胆的想法，还是渐进式的改进，这有助于帮你确定提出哪种想法。
- **要改变的事情** 你要带着自己的想法，想在公司改变或实现什么。了解用户不满的重点会给你一个很好的起点。
- **你为什么想要这份工作** 你应该充满激情地说出自己为什么热衷于这家公司，以及为什么热衷于产品管理。

- **为什么你会是一个优秀的人选** 同样，你应该可以抛出具有说服力的论据，证明你个人的技能和背景确实匹配这个职位。职位描述是很好的开始，但你要更深入地思考。公司面临什么挑战，你为何适合解决它们。

面试能让你对这些因素有更深入的了解，但不应该依赖于面试。应该尽可能先了解相关信息，再去参加面试。

10.5 问题

做好所有这些研究之后，你应该准备好参加面试时询问面试官的问题。几乎所有面试官都会给你提问的机会，你肯定不想到时候出现“尴尬的沉默”情形。

可以主要考虑以下几类问题。

- **实用的问题** 当被问到“你有什么问题要问吗”的时候，当然，你应该抓住这个机会了解下这个岗位或公司。这时候问一个开放式问题会特别有价值。比如说，“您觉得在这里做产品经理最有挑战性的是什么？”
- **展示激情的问题** 有些问题会展示出你对公司或岗位的激情，只是因为你的背景或提问的问题。比如询问公司为什么要放弃他们最初的商业化策略，这表明你已经研究过公司了。再比如，“你认为贵公司 5 年内会发展成什么样”表明你有了解公司的愿望。
- **专业问题** 其他问题不仅表明你做过研究，还能表明你对公司业务的深刻理解。比如下面这个问题表明了你对国际市场的理解：“如果可以分享，我想知道贵公司为什么决定在进入亚洲市场之前先进入欧洲市场。考虑到欧洲的法规问题，以及欧洲市场规模较小，似乎更应该选择先进入亚洲。”

除了这些常见的问题，还有些在很多面试中可以问的问题。下面是一些不错的问题。

- 你的一天一般是怎么度过的？你花多长时间写规范，跟设计师合作，以及做其他事务？
- 产品经理的岗位有哪些变化？你怎么看待这些变化？
- 产品经理、开发人员和设计师之间的关系是怎样的？决策机制又是怎样的？
- 在这里上班，你喜欢负责哪块儿工作？
- 成为你心目中理想的产品经理候选人需要具备什么条件？
- 是什么让你觉得这家公司的企业文化是独一无二的？
- 你觉得在这里做产品经理最有挑战性的是什么？
- 你在核心团队与扩展团队中跟谁一起工作？

不要低估你向面试官提问的重要性，很多应聘者发现面试时有大量时间可以让他们提问，但他们就是没有足够的问题可以问面试官。

不要问什么

不能问的问题如下。

- **危险信号** 比如说，在面试中问跟假期相关的问题完全不符合。
- **显而易见的** 询问一个答案显而易见的问题说明你没做好准备，可能对这个岗位也没太认真对待。比如说，问“商业化策略是什么”大多数情况下表明你的准备不充分。
- **批判式的** 比如“你们为什么没做……”之类的问题有时会产生负面影响，或者是那种意图证明自己很聪明的问题。这种问题可能会招人烦，特别当问题的答案可能是“要完成的功能太多，给的时间太少”时。

如果担心某个问题可能会暴露你的弱点，等你收到录取通知后再问也不迟。

第 11 章 定义自我

面试时可能会问到的问题范围非常广泛，但有 6 个问题出现频率很高，值得特别投入时间研究，你应该提前准备好答案。也就是说，你不仅要知道自己将要说什么，还要排练这些答案。

11.1 “介绍下你自己。”（个人经历）

很多面试官都会用开放式问题“介绍下你自己”作为开场白。实际上，经过一整天的面试，你一定会碰到这个问题，所以你应该准备好一个扎实的经历来讲述你的背景、工作成果和兴趣。

这时候你不能只是照本宣科地念你的简历，或者唠叨你的个人生活，而是要选出几个想让面试官了解的重点。同时，这也是一个将你的背景和应聘岗位联系起来的机会。

看看这个经历。

面试官：简单介绍下你自己吧。

应聘者：

好的，我很乐意。

我现在是 MapSign 的产品经理，已经做了两年了。

我进入产品管理这一行纯属偶然，当时我发现自己愿意跟客户打交道。很多人觉得技术会把他们的生活变得很复杂，但我觉得不一定非要那样。我喜欢聆听客户的需求，了解他们究竟面临哪些问题，并想办法让技术简化他们的生活。

我大学毕业后，到一家小型创业公司 MealRight 做软件工程师。我们做得不错，只是没能实现快速增长。

我家就是做餐饮业的，所以我对餐饮业有点儿了解，我建议改变工作重点，让餐厅加强对供应商的了解。我拜访了很多餐厅来证实这个问题，并成功地说服团队改变了方向。

> 几年后我做了产品经理，领导一个有4名产品经理的团队。
>
> 我现在在MapSign做产品经理，管理个性化功能。
>
> 我最近推出了一个功能，让用户可以创建自己的地图。实际上这是我在一次参加黑客马拉松时创建的，但事实证明这个怪念头得到了用户的喜爱。他们可以用这个功能“讲故事”。之前用户使用我们的产品大部分都是私人行为，现在人们会跟自己的家人和朋友分享他们的地图，这让我们的付费用户增长了15%。
>
> 如果你愿意听，我可以更详细地介绍我在MapSign的工作。
>
> 工作之外，我会出于兴趣写点代码，参加我刚才提到的黑客马拉松比赛。我还经营着一个以纽约市乐坛为中心的博客。管理由10名写手组成的团队——主要是大学生，他们乐意免费写些文章。我们每个月有20万左右的访问量。
>
> 我现在希望能在一个处于早期阶段的公司工作，能把一个产品从概念设计阶段一直做到发布阶段。曾经所拥有的开发和产品经理跨职能经验让我觉得可以胜任这种工作，甚至可以承担其他几个岗位的一些职责。这个职位有很多吸引我的地方，对我来说现在进入是最好的时机，并且它和我眼下正在做的个性化工作有密切的关系。

在这段经历中，应聘者带着面试官回顾了一下他的背景，同时在叙述中还穿插了他的主要成就，把不同的工作成果连在一起，并鼓励面试官在他擅长的领域中提更多问题。围绕着特长以及喜欢做产品经理的理由，一个完整的故事就形成了。

设计讲稿时，要想一想你希望面试官了解哪些背景、经验和兴趣。如果可能的话，把经历中的要素跟公司要寻找的东西连起来——不管它是产品经理岗位职责所特有的，还是从公司产品中体现出来的。

此外，要牢记这些必须做的和不能做的。

- **一定要**知道自己讲了多长时间。面试官会评判你的沟通技能，没人会喜欢一个话匣子。如果面试官做了自我介绍，那你的介绍时间最好是她的一到两倍。如果没有，最好控制在2分钟左右。
- **一定要**突出你的工作中最有意思的或者相关性最强的地方，这是你推销自己的机会。
- **不要**只是列出你的工作成果。那样听起来就像是在自夸一样，并且面试官在面对所有细节时可能会觉得没有逻辑。你应该完整地介绍下你的成长经历，在讲述时要把生活中的不同要素串起来，同时要让面试官能从你的过往背景中了解到你适合该职位的理由。
- **不要**过于强调技术。有非常好的技术经验确实很棒，但面试官想听你用更直白的语言讲解那些工作为什么重要。毕竟，作为产品经理，你不仅要跟懂技术的人沟通，同时还要跟非技术人员沟通。
- **一定要**提到相关的“业余”活动。比如说，如果你应聘一家跟健身相关的创业公司，并且你曾发起过一个马拉松训练小组，就应该提一下。那表明你对这个领域有激情，并且可能是这一行的专家。即便这些业余活动与应聘看似没有直接的关系，但它们一般也能

展现你的主动性和领导力。

- **一定要**练习。几乎肯定会问到你的“经历”，如果没准备就尴尬了。拉个朋友讲给他们听，听完之后再问问他们的意见？你可能甚至要自己录下来，这样你自己就能知道听起来如何了。
- **不要**讲得太抽象。你不应该说自己做了客户研究，写了产品规范，应该讲一些你学到的重要经验，以及你怎么基于那些经验改变了产品设计。
- **不要**搞得很沉闷，只是背一堆关于自己的事实，而是要把你的经历编成一个小故事。
- **一定要**充满激情，为自己过去的工作感到自豪。

你应该正视这个问题，这是你向面试官推销自己为什么是完美候选人的机会，一定要提前作好充分的准备。

11.2 “你为什么想来这里工作？”

我在谷歌面试应聘者时，会问他们为什么对那个职位感兴趣。大部分答案可以归结为下面这几条：

- 因为我听说谷歌的企业文化很好；
- 因为谷歌改变了世界；
- 因为有那么多人使用谷歌的产品，如果能改善或增强家人和朋友们现用的产品，那就太好了。

这些答案没问题，但它们也仅仅是没问题。给出这样的答案你得不到任何加分。为什么？因为你的答案太平淡无奇了。

理想的答案会以某种方式推销你，试着将下面这些因素中的一个或多个融入到你的答案中。

- **研究公司**　你可以在自己的答案中表明你已经研究过公司或这个职位了。做研究说明你对职位的激情，有激情的员工会变成好员工。比如，你可能会说：“我的兴趣源自你们一位首席 UI 设计师做的报告，其中演示了你们做数据分析的不同方法，以及这些工具如何帮你们推出更好的产品。我非常推崇量化，所以我非常想在这样一家由数据驱动的公司工作。”
- **相关经验**　你的答案实际上能直接把你的相关技能或经验传达给面试官。比如，你可能会说：“我对测试工具很感兴趣。在大学期间，为了做好某个项目，我使用了一种自动化方式来检测某类错误。我了解到了各种各样的网站错误及其检测方式。如果处理得当，自动化测试所产生的影响真的让我着迷——不过我也认识到想要做好有多困难。所以，我很高兴能学以致用，重新从事这一领域的工作。”
- **激情**　能直接传达出对一个职位的激情非常有价值。对于那些专注于让世界“变得更好”

的创业公司尤其是这样。比如，对于一家与教育有关的公司来说，可能像这样的解释会比较好：“我小时候家里很穷，上的小学非常一般。我之所以能出类拔萃，赢得顶尖高中和大学的奖学金，是因为找到了额外的资源，那就是一些公共图书馆和少量的几家网站，另外还得益于一些热心的良师益友。我觉得你们公司完全可能将这些线上资源带给更多处于弱势群体中的学生。因为这些因素曾经对我的成长非常有益，所以我现在真的想成为其中的一份子。”

一般来说，大公司对应聘者会采取比较中立或保守的态度，即便你研究过那家公司或激情澎湃地想要在那里工作，但很难一下子打动他们。但不管怎么说，他们通常都会认为你的应聘代表你真的想在他们这里工作。最好能够在应聘时展示自己对岗位/团队的激情，或者一些能有说服力的经验。千万不要说这会是你职业生涯中的垫脚石，或者会让你的简历更好看，那些答案可能会让面试官觉得你在他们那里呆不了多长时间。

在创业公司，对公司的激情或在那个领域的专业性十分重要。你应该使用他们的产品，并且要能有自己的想法和观点。回答时不要提及将来的“红利”，那会让你减分，因为这可能让别人觉得你只在乎钱，还说明你可能没意识到创业的艰辛，或者可能会在艰难时刻选择离开。

尽量在面试前想好两到三个原因。当然，要练习你的答案。

如果有多个面试官提问了你这同一个问题，跟每个人讲的都差不多也没关系。实际上，如果你给出的原因不一样，就会显得很可疑。

11.3 “为什么要雇你？”

这个问题显得有点气势汹汹，但类似方式的提问也有很多，比如面试官可能会问你：“为什么你觉得自己适合这个岗位？”或者“你觉得自己能为公司做些什么？”

尽管这个问题显得很恐怖，但实际上这是个好问题，你可以借此时机推销自己，而不用再费尽心思地穿插些小花絮来彰显自己。

对于这个问题，答案可能是下面这几项或其中某个。

- **为什么你是一名优秀的产品经理。**你在当前工作中展现自己的主动性了吗？你有深厚的技术技能吗？你在过去是一名成功的产品经理吗？可能的话，拿出证据来会更有说服力。
- **为什么你适合这个领域。**你曾经在与团队或公司相关的领域工作过吗？你真的热衷于这个领域吗？为什么？你可以抓住这个好时机介绍自己对公司或团队所在行业的了解。比如说，如果你对广告行业很了解，跟面试官讲讲你在该领域的经验。
- **你为什么适合这个公司的文化或工作环境。**有时候，你的背景很适合该公司独特的环境，这也是个不错的加分项。比如说，在一家小型创业公司内，产品经理要承担的职责远远

> 多于传统意义上所应承担的责任。可能你会说:“我喜欢承担多种职责，在工作之外，我曾发起过一个志愿者组织来教导儿童，从而有机会接触到了营销、广告、活动策划、管理和招募等各种工作。我发现自己乐于身兼数职，并且不怕担负重托。”

对公司或职位要求的了解自然是越多越好。在面试之前，重新看看职位描述，还可以找几个匹配的案例，了解公司在寻找什么样的人。

回答“介绍你自己”以及“你为什么要在这里工作？”所用的答案中，有很多元素都可以用来回答这个问题，但你要更直接地使用这些元素。

对于这个问题而言，下面这位应聘者的回答无疑十分理想。

> 我觉得我之所以适合这个岗位，主要有三点原因。
>
> 首先，我做过几年产品经理，并且做得很成功。任职期间，从无到有推出过四个重要功能，并被评选为公司里的顶级产品经理。我目前就职公司的招聘经理就是因此把我招到了现在这个团队。其次，我主修的是计算机科学，并且还辅修了统计学，符合你们要找的具备技术和量化背景的人选。我在当前岗位上还没做过大量数据分析工作，但我有学术背景，并且有良好的学习能力。实际上我已经通过一些网上教程开始学习数据分析了，我相信自己很快能重新获得岗位所需的这些技能。最后，我真的想加入你们。我是个健身迷，跑过三次马拉松，参加过两次强悍泥人障碍赛。我认同你们将运动游戏化的使命，并且也一直在想办法让运动变得更有趣。我曾经招募过几个对抗练习的朋友定期锻炼，并且乐在其中。

这位应聘者的回答中有几处值得我们学习借鉴的地方。第一，他表示了自己想要在那里工作的意愿，并且表示自己能很好地胜任工作；第二，针对面试官可能会觉得自己在数据分析专业上有差距这一方面做了补充；第三，利用这个机会，他列出了没能在简历上展示的具体的、实在的成就，比如被一个经理招募的经历；第四，整个对答条理清晰，组织良好。

由此可见，用一种逻辑清晰、思维缜密的方式回答这样一个开放式问题，不仅能向面试官证明你善于沟通，还有利于让面试官记住你给他传递的信息。

11.4 “为什么要离开现在的工作？”

一般而言，这个问题是当破冰船来问的。面试官不一定希望得到什么具体的答案，他只是想进一步了解你，而你的基本目标就是别把事情搞砸了。

什么样的答案会把事情搞砸呢？下面这些任何一个都会。

- **抱怨你当前的工作** 面试官不喜欢消极的应聘者。他们担心应聘者会一直消极下去，甚至在被招进来之后依然如此，或者他们可能会觉得：应聘者之所以不喜欢当前的工作，

实际上有自身的原因。如果应聘者陈述跟当前的老板关系不好，那么有可能他自己也有部分责任。

- **太看重钱** 一个只是想赚更多钱的应聘者通常不会变成一名优秀的员工。面试官会担心这样的人难以对工作产生激情，有所担当，而一旦能赚到更多钱时，他们可能就会离开公司。
- **厌烦，不爱学习** 虽然有很多积极的方法可以扭转这个局面，但厌烦当前的工作说明你大部分近期从业经验都没有多大价值，或者没什么意思。相信你不会想让面试官产生这种念头的。

相反，应该强调积极的因素，要把你真正想要在新岗位上寻求的东西表达出来。

下面这些从良性到积极的原因，都可以成为你解释离职原因的措辞。

- “老实说，实际上我在现在的岗位上做得很开心，真没想过要离开。不过我偶然发现这个职位空缺，看起来真的是一个不能错过的好机会。”
- “做了几年的开发经理，我积累了深厚的技术和领导经验。现在我想带着这些技能进入产品管理领域，因为我真的喜欢推出新产品的经历。”
- “完全是由于个人原因。”
- “我们公司去年被收购了，所以发生了很大变化。我想回到更有创业精神的环境中。”
- “实际上我真的很喜欢现在的公司和岗位，但在那里呆了几年之后，我想到一些能更多回馈这个世界的地方去工作。”
- “我有很深的统计背景，并且我很喜欢那种量化的感觉，所以我想到那些能更加直接使用这些技能的地方。”
- “现在所在的团队更关注企业，而我想更接近消费者。”
- “我在现在的岗位上得到了很多好机会，但呆了 3 年之后，我觉得学到的东西渐渐变少了。我们公司和整个行业都没有太大的发展，我想应对新的挑战。”

不用说任何废话，要讲得简短动听，一到三句话就足以解答这个问题了。

11.5 “你在业余时间喜欢做什么？”

问这个问题，面试官主要是想了解你和你的兴趣。回答这个问题时最好展现一些跟岗位相关的经验。不过，谈论一下工作之外任何让你有热情的事情，也能起到一些积极的作用。

11.5.1 最佳答案

下面这些答案跟岗位有关，突出了应聘者的领导能力、技术天赋或主动性。所有这些都能说明应聘者适合做产品经理工作。

- “我非常喜欢探索新技术。我确定过一个目标，用尽可能多的语言做一个简单的待办事项程序，只是为了了解一下每种编程语言要做什么。到目前为止，我已经用过 5 种语言了。”
- “我最近在接触在线学习。我参加了一些在线课程，为的是了解各种岗位，比如营销、广告和财务。我还准备推出自己的课程：面向软件工程师的产品管理。”
- “我为动物收容所做了很多志愿工作。几个月之前，我创建并推出了一个新计划：Teens 4 Animals（青少年与动物），为的是让更多的青少年能参与进来。这项计划从当地高中招募了 50 名新志愿者，拯救了 100 多条将被处以安乐死的狗。”

11.5.2　较好的答案

下面这些答案中没有跟产品经理岗位特别相关的内容，但确实展示了一些工作之外的兴趣。更重要的是，它们都有实际的证据支持，从而让这些兴趣听起来更有说服力。

- “我主要是读书。最近我读了很多心理学方面的书，我十分痴迷于了解思维的工作原理，以及我们为什么做出那些选择，很多理性分析和我们的直觉都不一样。”
- “我喜欢玩极限飞盘。我在我们公司的飞盘联盟里玩，还参加了我们社区组织的一个联盟。”
- “我喜欢在我的房子周边做基础建筑项目。我还要学很多东西，但已经取得了一些成果。几个月前，我把露台重新做了一下，现在它还没塌。”
- “我是一个朋克摇滚乐队 Out Like Pluto 的主唱兼主要的词曲作者。我和一名乐队成员合作创办了这个乐队，并且我还负责大部分社交媒体营销。”

11.5.3　没有附加值的答案

要想把这个问题搞砸了真的很难，但有很多答案对你不会有什么帮助。下面这些答案之所以平淡无奇，是因为它们没表现出任何激情。你的业余时间就是上班、回家、陪同家人和朋友。这种答案没什么伤害，但却使你错过了拓展职场空间的一个好机会。

- “我不太确定。我猜我真的没什么爱好，因为我在工作上花了很多时间。”
- “我大部分情况下是跟朋友混。我们去吃饭，有时候还会品酒什么的，偶尔我在家里组织大家一起烧烤。”
- “我花很长时间在网上看帖子、博客、技术新闻之类的。”

大部分人确实有爱好，如果他们仔细想想的话，还能拿出证据证明自己的爱好。你平时锻炼吗？给自己设定一个目标，比如为了马拉松训练，或者每个月开发十个新的锻炼套路。你是不是在网上花了很多时间，通过写博客或者在论坛上回答问题高效地利用那些时间。你喜欢摄影吗？上摄影课，或者设置一个在线影集，即便水平一般也没关系。

不要试图把你所有的业余时间都体现在这个答案里，只要让你的档案更加丰富，并令人信服地谈论它。

11.6 “你对自己在未来 5 年里有什么规划？”

这个问题会吓坏应聘者，因为他们觉得面试官可能想给他们出难题，然而事实不是这样的。

面试官问这个问题一般是出于以下几种原因。

- **想要知道你是否真的想要这份工作。**如果这份工作跟你的长期目标不一致，那面试官会觉得你把这份工作当成了一个短期过渡。此外，如果你没准备好回答这个问题，就表明你实际上对这份工作并不关心。
- **想看你是否有规划。**成功的人往往知道自己的生活要向什么方向发展。如果你没有规划，面试官可能会担心你对自己的职业生涯不够严肃。
- **要测试你的野心。**有野心的人一般会成为好员工。如果从你的长期目标来看缺乏这种野心，那可不是个好现象。同样，如果你有不切实际的野心，也不是好现象。如果公司无法满足你的期望，那你也不会愿意很好地融入。
- **确保公司能满足你想要的。**如果一个岗位无法匹配你的长期目标，公司希望事先能有所了解。你对工作不满，就不太可能会成功，并且可能很快离开，而这绝对不是一个好现象。

理想情况下，你会有 5 年、10 年和 20 年规划，并且这个岗位能很好地跟它保持一致。

如果你没有做过这样的规划，可以换个角度来思考这个问题，往往答案就在其中：在这个岗位上特别强劲的表现能让你在 5 年里发展成什么样？这也可以是你希望看到的自己的未来。

如果事先准备过，这个问题会成为一个很好的展示机会。你可利用用它告诉面试官你自己想要达成的任何目标。

> 我想要成为新生业务单元的首席产品经理，我将在那个岗位上考虑长期战略，特别是跟盈利模式相关的。我对移动端非常感兴趣，我应该能带领着一队产品经理和工程师去开发移动端产品。此外，我对帮助新员工快速发展的方法也很感兴趣。我应该会创建出一个更加正规的新员工培训计划，使员工受到不同业务领域的指导和训练。

在这个答案中，应聘者既展示了自己的野心，也展示了自己对业务模型、领导力和指导的激情。

在回答职业生涯愿景的问题时，可以用自己的答案突出你擅长什么，以及什么最能激励你。

11.7 “你有哪些优点和不足？”

尽管很多面试官已经不问这个问题了，但它还是很常见，你也应该做好准备。要为面试准备好三个优点和三个弱点。

要把你的优点集中在具体的、跟岗位相关的地方，并且要有证据支撑。比如，你可以这样说：

- “我主动性很强。我不怕动手尝试或承担风险。在我的上一份工作中，我感到如果能让很多非工程人员了解产品是如何构建的，将是一个很好的事情。所以我发起了一个周计划，让工程师给非技术人员讲解关于产品的技术、技术最近的发展，以及其他相关的主题。公司里几乎每个人每个月都至少参加过一次这个技术会谈，有 10%的同事参加了所有的会谈。”
- “我非常喜欢找到做事的创新性方法。比如说，我想在大学里教一门课，而没毕业的人是不允许那么做的。后来我找到一位德高望重的教授，由他正式督导，但并不直接教学，这样实际上我自己把全部教学都包了下来。”
- “我觉得自己最大的优势是有趣、不固执。我会对自己的想法充满激情，但也不怕承认自己是错的。这样当产品或市场迅速发生变化时，我也能很灵活，这样同事们也慢慢地更加相信我了。”

不要把“智商”当作优势。是的，你可能非常聪明，而且，智商确实和这个岗位有关系。然而，这样可能会显得傲慢，而且坦率地说，你是否聪明由面试官说了算。

至于弱点，你要给一个真正的弱点。过去那种把优点说成缺点（“我工作太刻苦”）的把戏老掉牙了，而且可能再也不管用了。面试官想要看到能够用于承认自己真正缺陷的应聘者。

那就是说，强调（或者至少是提及）弱点积极的一面是个好主意。你从中学到了什么？那有什么好处吗？你是如何弥补这个弱点的？

- “我经常会忽略细节。尽管这有些好处，特别是在创业环境中能让我动作更迅速，但也让我犯过一些尴尬的错误。不过我已经认识到自己在哪些情况下可能会犯错，因此在那些情况下我会再次，或者第三次检查我的工作。”
- “我有时候会变得太消极或苛刻，即便有时那并不是我的本意。我知道自己过去伤害过一些人的感情，特别是在我职业生涯的早期。我已经花了很多努力来改善自己的沟通风格。对于任何负面反馈我都会给出积极的评价，并且我用提问题取代了直接批评。”
- “我有时候工作时容易分心，特别是有很多不同任务要做时。我尝试过一次性把所有事情都完成，但负担太重，而且都没有什么实质进展。当我承担多个竞争性职责时，我发现可以通过做一个清晰的待办事项列表来解决这个问题。这让我工作时结构性更强，也防止我出现不堪重负的感觉。”

如果它让你觉得更舒服，或者能帮你强调积极的地方，你可以将自己的弱点说成“我要解决的一个问题是……”这表明你正在克服自己的弱点，并将它转向一个你不介意谈论的话题。

一般来说，试图对弱点避而不谈太战术性，或者说只能是一时的。“我在人员管理上没太多经验”这样的答案对面试官了解你没什么帮助，也显示不出你的谦逊。实际上，只会显得你不敢正视自己的弱点，或者认为自己性格上没有什么缺陷。

当然，应聘者也可能会说出对自身非常不利的弱点，比如不诚实或职业道德低下，不过这很罕见。如果应聘者在这个问题上不过关，那么问题肯定出在他们给面试官留下了一种试图掩盖真实弱点的印象。

下面我们提供了一个优缺点例子的列表。这只是为了抛砖引玉；你的优缺点可能不在这个列表里，应该找到适合自己的。

11.8 优点的例子

善于分析	斗志旺盛	创造性
精力旺盛	有条理	果断
思考时不受条框限制	敢于冒险	能承受压力
缜密	看事情透彻	理解人的感情
灵活	主动	注重细节
善于规划	定量	多任务
领导力	善于接受反馈	持久
有说服力	数据驱动	独立
严格要求自己	好导师；关心人	不畏惧挑战
主次分明	喜欢学习新技能	能给团队带来欢乐

11.9 缺点的例子

不注重细节	过于自信	缺乏自信
太消极	做太多的假设	不切实际
不自信	缺乏耐心	优柔寡断
固执	胁迫他人	拖延
接受反馈时会掺杂个人感情	难以承认失败	不愿寻求帮助
太直接/生硬	过度分析	爱争辩
容易分心	会非常空泛	不善于多项任务
微观管理型	注意力时间短	害羞

第 12 章 行为问题

行为问题多种多样，范围各异。招聘者可能会问你如何应对某个假设情况，或者如何处理真实的特定情况。可能还会要求你讲解简历上的某一部分内容。不过他们最终要考察的只是两个因素：内容和沟通。

可惜很多应聘者都会忘记其中的某个因素，甚至把两个都忘掉。他们只是尽量给出对应的答案，却没能关注到答案的隐含信息，或者回答的方式。

因此你的目标就是两手都要抓，两手都要硬。

12.1 为什么会问这些问题

前面说过，行为问题考察的是内容和沟通。这其中包含很多因素。

12.1.1 内容

在评估你回应的内容时，面试官可能会就以下几个方面深入研究。

1. 你真的做了你说的那些事情吗？

面试官之所以会问行为问题，一个重要的，并且经常被忽视的原因是要验证应聘者的简历跟他的经验是否一致。这并不是说面试官以为你在撒谎（尽管确实有人会那么干），而是因为他很难仅凭 15 个词的要点评估一个人过去做了什么。

比如一份简历中有这样的要点："通过实现新过程使用户留存率提高了 15%。"

这看起来相当不错，但它是真的吗？新"过程"可能只是在电子邮件中加入一个续约链接。当然，这种做法也挺不错，但相对来说是种侥幸的偶然事件，并不能保证你将来就一定能取得这样的成功。

换句话说，那个改变可能并不像你所描绘得那么动人。仅凭一两行字，真的没法把某些事情的困难之处解释清楚。实现一个新过程也许需要打造一个临时团队，采集数据，并进行复杂的分析。

实际上面试官也不能肯定他们对简历的解读跟你的真实经历是一致的。你取得成就的过程可能要比简历上表现出来要艰难得多，也可能容易得多。面试官可以通过行为问题“深入挖掘”你之前的经历。

2. 你是怎么施加影响的？

状况的规模大小也就表明你的成就大小，以及你对成就的认识。

比如说，假定面试官问了你这样一个问题：“讲一个你曾经遇到过的比较有挑战性的问题。”

实际上这是一个非常开放的问题。你面临的挑战可能是团队协作问题，或者数据分析或量化问题。甚至可能是个人问题。

面试官会注意到你所选的答案的规模和类型，那会透露出很多与你职业生涯及成就相关的信息。涉及成功、失败和挑战的问题尤其是这样。如果你在专业上没有什么明显的成功和失败，很可能表明你实际上做得并不太多。

3. 你有跟工作相关的技能和特点吗？

除了以上那些要素，面试官一般还会用行为问题评估你的技能和特点。面试官可能在脑子里想着一个非常具体的技能或特点，比如领导技能；也可能会问及一些非常宽泛的问题，以便了解你处理问题的方式。

比如要求“讲一个你就某事说服经理的经历”，这是要看你在没有权力时为何要以及如何影响别人。

- 是不是经理说什么你就做什么？如果不是，你什么时候会提建议，为什么？是关系到你的个人目标或团队中的那些目标吗？是关系到短期或长期利益吗？
- 在你没有权力时，你是怎么影响别人促成某事的？你会收集数据来支撑你的结论吗？你会争取身边人的支持吗？你会引入风险或机会吗？你用什么工具影响你的经理，其中哪些成功了？你是怎么适应经理的风格的？

其他问题可能更宽泛。比如说：“讲一个你犯过的错误”很可能是在考察很多其他因素。面试官通过这样的问题了解你如何处理状况，如果面对的是真正的困难，那么你又是如何克服一些错误的，以及你对错误的判定。这些问题还能向面试官表明你是否谦虚，以及当把事情搞砸时，你是否勇于承担责任（希望如此）。

12.1.2　沟通

行为问题有一些讨巧之处：即便你不能搞定内容部分，至少也可以搞定沟通部分。其中的技巧在于使用某种特定的话语组织形式。

对于还在为沟通问题挣扎的人，包括英语不是母语，以及容易东拉西扯的人，这个建议特别重要。

1. 首先要点题

“首先要点题”是个很简单的话语组织形式，也就是说你要从所讲故事的“主题”或论点开始。

比如说，当面试官问你工作中面对的挑战时，你可能会这样开始：“好的，我给你讲一个表现不佳的同事的事情。”

有了这样的开场白，面试官就能把注意力集中在你要讲的内容上，进而在这个背景下对你所传达给的所有信息进行考量，因此剩下的内容也显得更有条理。当你提到更多细节，比如当你提及项目到截止日期时还没完成，面试官也能知道内容重点在于你那位表现不佳的同事，而不是超出截止日期。

这也能有助于你能专注于讲述自己的内容，因为你知道自己要讲的是表现不佳的同事，所以不太可能牵扯到一些无关的细节。

2. 状况、行动、结果（S.A.R）

“状况、行动、结果”这个组织形式既可以独立使用，也可以跟“首先点题”结合起来用。

S.A.R（还经常被称为 S.T.A.R——“状况、任务、行动、结果”）通常包含以下几个部分。

- **状况**　这里的目标是提供充足的背景信息，让面试官了解你在做些什么，以及为什么要做。要注意，别给面试官太多不相关的信息。比如说，如果要介绍一个你们团队的项目，可能没必要讲项目的细节。只要说：“我们正给一个重要客户做个项目”就足够了。
- **行动**　然后介绍你采取的行动。注意主要集中在你做了什么上，而不是你的团队做了什么。毕竟参加应聘的是你自己。
- **结果**　解释行动的最终结果。你是如何帮助团队或公司的？其他人有什么回应？如果可能，要把你的影响量化一下。最好能告诉面试官你所采取的行动导致用户保有量增长了10%，而不能简单地用“更大的用户保有量”来形容。

假定面试官会问你：“讲一个你必须对团队施加影响的经历。”你可能会讲这样一个故事。

- **主题**　好的，我来讲讲有一次必须缩减项目范围的事吧。

- **状况** 为了赶在一个重要的日子之前发布产品，我们需要砍掉一个功能。不幸的是，开发人员特别看重这个功能。如果我只是跟他们说："我感到非常抱歉，但这个必须砍掉"，那会打击他们的士气，而且还会让他们觉得自己的看法不被重视。
- **行动** 于是我找了两位特别有影响力的开发人员，其中一位是有着深厚的资历，另一位则深受大家爱戴。我必须先争取让他们支持我的决定。

  我先找到那位资深的开发人员，跟她讨论即将到来的截止日期，以及这对产品架构的影响。我们讨论各种可能的选择及后果，并最终达成了一致：必须砍掉这个功能。

  之后我又找到另一位开发人员，跟他也说明了情况，不仅如此，更多从情感。我知道他清楚自己带给团队的影响力，所以我告诉他，他的支持对我有多么重要。他跟我讲了他对团队反应的担心，不过，他保证一定会支持我。

  最后我请他们两位帮我一起向团队传达这个决定。
- **结果** 最终团队虽然对这个结论感到失望，但也表示同情。听到他们高度重视的两位团队成员支持这一决议，他们相信那是正确的选择。我对他们承诺，如果我们能尽快推出这一版，发布之后就会接着做这个功能。 最后我们赶在截止日期之前一天完成了任务，然后在几周之后实现了那个功能。

用这种方式组织答案，面试官可以清晰地了解进展情况，你采取的步骤以及结果是什么。需要注意的是，在这个例子中没必要谈论具体的功能，或者为什么开发人员那么重视它，因为这些跟整个故事的主题无关。

3. 按要点讲话

有时人们会开产品经理的玩笑，说因为他们总做 PPT 报告，所以讲话都是按要点来的。这并不是坏事，实际上这非常好。

在你跟面试官（或者任何人）谈论你的经验时，他们对相关背景的认知跟你有很大差距。他们很容易搞混很多细节。如果面试官对故事中的目标、任务或结果有非常清晰的认识，就更容易掌握你讲的故事。

比如说：

- "我做了三件事。第一件事，我跟……谈话；第二件，我……；第三件，我……"
- "我们这个计划有两个问题：第一个问题，我们……；第二个，我们……"

如果你能把自己要说的内容组织成要点，那就按那种方式讲，面试官就可以更加轻松地获取你要传递的信息。

12.2　准备

要给出强有力的内容，你就需要好好准备。实际上，不管从哪方面来看，针对行为问题做好准备，结果可能会事半功倍。当你知道自己会被问到某些行为问题，经过适当的准备，就可以很好地把握这些问题。

12.2.1　第一步：创建一个准备事项表格

你可以用准备事项表格将自己的经验对应到常见的行为问题上。大概是下面这种样子：

	工作1	工作2	业　余
领导力/影响力			
团队协作			
成功			
挑战			
错误/失败			

把你简历中的每个“主干”部分做成表中的一列：每个岗位、每个项目，以及每个业余（志愿者等）活动。表中的每一行对应行为问题的一大类：领导力/影响力、团队协作、成功、挑战，以及错误/失败。然后每个单元里填上一到三个故事。

有了这样的表格，你就很容易从中找出针对特定问题的答案。比如当面试官问道：“讲讲你是如何应对一个有敌意的同事的？”或者，“你在上一家公司是如何表现自己的领导力的？”就可以直接提取一个故事讲一讲。

12.2.2　第二步：掌握 5 个关键故事

上面的单元格里有很多故事，但你要掌握的是最能代表产品经理胜任程度的 5 个关键故事，那些你一有机会就准备宣讲的故事——它们必须要展现出你个人的杰出能力。

在自己过去的经历中确定这 5 个关键故事时，可以考虑下面这些问题。

1. 每个故事都有实质性内容吗？

每个故事都应该有实质性内容，包括状况、行动和结果。大多数情况下，故事会在“行动”部分出问题。

一位应聘者跟我进行了一场模拟应聘，下面就是她说的故事：

> 我的团队刚推出了一个新功能，让用户可以在移动应用上更新自己的档案。起初一

切都很正常，但不久后，就有几个用户开始抱怨他们收到了很多垃圾信息。

我把开发人员召集到一起，要求他们集中全力检查一下，看是否因为我们的问题造成的。最终证明，问题源于软件在更新过程中不小心修改了用户的隐私设置。

所幸，开发人员可以修正这个问题，他们回退到老的隐私设置上，并推出了一个补丁。

然后我起草了一封邮件，向用户解释这一状况和我们的补救措施。用户对我们的解决办法感到满意，并且这个问题再也没出现过。

应聘者讲完这个故事后，我点点头，说道："所以，基本上，你只是写了封邮件，是吗？"

"哦，是的，"应聘者告诉我，"但如果不是因为解决得快，这可能真的会成为一个大问题。"

她说的没错；它可能会。但这个麻烦是开发人员解决的，这位应聘者所做的只是写了封道歉信。那不是实质性内容。

要确保每个故事都包含"过硬的"状况、行动和结果。找个人跟你一起排练一下，让对方重复你讲的这三部分要点。

很多一开始出问题的故事都可以完善。比如说，如果这个应聘者整合了一个新的流程来防止这些状况再次发生，那这就变成了一个好故事。

2. 每个故事都容易理解吗？

如果故事过于复杂，不管有多好的状况、行动和结果都没用。面试官最终会被不明确的分工搞糊涂，没办法抓住最主要的部分："你做了什么?"

当然，好在你可以把故事讲得容易理解。要知道每个故事的重点是什么，这样就可以只提供相关部分的内容。

如果你已经将故事压缩到了最基本的部分，但仍然发现自己要解释很多细节，那可能应该放弃这个故事。这种情况在那些需要技术、产品或数学知识的故事中很常见。

3. 每个故事都提供了你的哪些信息？

你给面试官讲的故事不应该只是回答问题，而是应该向面试官传达一些关于你如何工作的信息。在考虑可能会要讲到的故事时，问问自己：这故事跟我的工作方式有什么关系？

比如下面这个故事：

在上一家公司，我找到一种方法，只需稍作调整，就可以支持一个新的细分市场——创业公司。然而 CEO 不愿意追逐这个机会，他担心新的市场回报不足以负担额外的开发投入。归根结底，他只是不愿意为了这么低的回报冒这个风险罢了。

不过我并未放弃。我找到一位核心软件架构师，制定了一个计划来测试这些新的调整。我们能把最小可行性产品的开发周期从 6 周缩减到 2 周，同时还找到一种以最小工作量支持这个市场的办法，缩减了营销、测试和客服的成本。

我还更透彻地研究了这个市场。寻找这个市场已有的其他解决方案，了解我们的产品跟其他竞品的竞争态势。然后搜索了我们的客户群，发现大概 5%的客户实际上是企业客户。

当我再次找到 CEO 时，我的提案风险就小了很多。我们只要花两周的开发成本就可以将这个新版本推向已有的企业客户。这几乎不存在任何营销成本——至少在当时是这样的。而且，因为我们不是把一个新产品推向新用户，所以也能很容易撤销这些修改。

CEO 同意按照这个计划探索新的细分市场机会。最终我们取得了极大的成功。实际上，这让用户消费额增长了 20%。目前，这个新市场现在占据了我们总营收额的 15%。

这位应聘者展示了她的：

- **创造性** 她找到了一个新的细分市场；
- **野心** 在其他人可能已经泄气时，她仍努力去实现自己的想法；
- **理解别人** 她明白 CEO 担心什么，并找到了解决这些担心的办法，而不是跟他争辩对错；
- **数据驱动/分析** 收集数据来支持她的观点；
- **精打细算** 找到了降低成本的办法，并分析了一个对公司来说风险很小的市场。

你甚至可以从另一个方向来考虑这个问题。你想要展示自己的哪些方面，并如何通过实例来传递它们？

- 你有创造性吗？
- 你在自己的公司里受欢迎吗？
- 你善于建设团队和激励他人吗？
- 你善解人意吗？有同情心吗？
- 你是好领导吗？
- 你擅于分析吗？
- 你是否果断又有野心？
- 你敢于冒险吗？
- 你善于降低风险吗？
- 你是技术型的吗？
- 你善于影响别人吗？

这些领域中的每一个都可能是新故事的一部分。

4. 它真的是关于你的吗？

一位优秀的产品经理一定会把团队放到第一位。

你通常都应该这样做，但在面试过程中要另当别论。注意你讲故事的方式，如果你发现自己一直在说“我们的”和“我们”而不是“我的”和“我”，可能要调整下你的措辞，或者找个新故事。

这些故事应该是你的成功和失败，而不是你的团队的。你的团队当然是那些故事的重要组成部分（实施或者感受到影响的），但重点仍然要放在你的行动如何促成那些影响。比如说，如果你的团队调转了方向，就应该讲讲你做了什么来带领团队通过那个调整。

而这其中的行动应该是你的行动，即便结果是团队或你的客户所感受到的结果。

5. 你能“理解”别人吗？

理解他人是团队协作、领导力和说服力的根本，也是产品管理岗位的根本。你的故事应该表明你“懂得”别人。

很遗憾，有时应聘者的故事揭示出她缺乏对他人的洞察力。这通常表现在以下几方面。

- **说别人坏话**　是的，我们都知道有些人很难与之合作，但面试可不是谈论这个的时候。如果你对同事说三道四，面试官很可能会想是不是你才真的是那个难以合作的人。很多公司都不会在这上面冒险。与其在你的故事里抱怨别人，还不如指出他们的动机。你可以说：“另外一个产品经理想要为了即时收益进行优化”，而不能说这个人就是个白痴。
- **显得无助**　失败没关系；无助不行。如果你不能就某事说服别人，说清楚原因，不要批评。比如你可能这样说：“尽管我的提案很振奋人心，但高管和我最终认为在项目接近尾声时改变进程风险太大了。”

同情心也是产品经理岗位中的重要部分，你的故事需要把它展示出来，最起码别起反作用。

6. 你的故事全面吗？灵活吗？

行为问题一般会涉及 5 个领域，所以要确保你的 5 个关键故事能覆盖这些主题。这些主题一般是：

- 领导力和影响力
- 挑战性
- 错误/失败
- 成功
- 团队协作

每个主题至少应该有一个代表性故事。

理想情况下，应该有几个故事能覆盖两个或更多领域，因为你不知道自己会被问到什么。如果有问题涉及挑战性怎么办？或者如果某个问题没有明确主题怎么办？很多时候，可能你会发

现，关于“挑战性”的故事已经用来回答“给我讲讲你带领团队战胜某项困难的经历”这种明确要求领导力的问题了。

要确保你的故事既涵盖面广，又要很灵活，以防被特定的行为问题搞得措手不及。

12.2.3　第三步：练习

你知道自己会被问到各种行为问题，所以没有理由不做好准备。如果可能，找个朋友听听你的故事。然后请你的朋友重复他所理解的状况、行动和结果的要点。是不是每一部分对你和他来说都有实质性内容？如果他很难理解你的故事，那么面试官很可能也是如此。这也就意味着说你应该找个新故事，或重新细化现在这个故事。

12.3　后续问题

有所准备并不代表你的面试就能顺利过关。根据被问到的问题，你还要准备好下面这样的补充问题。

- 团队是如何应对某个问题的？
- 这将会如何影响团队的未来？
- 你从中学到了什么？
- 如果这种状况再次出现，你会做哪些不同的选择？
- 你总是像这样处理各种状况吗？如果不是，为什么这次用这种方式处理？

12.4　行为问题的类型

以下列出了一些常见的行为问题。不过这些问题的形式可能会有很多种，并且要具体得多。比如说，面试官可能不会问你曾经面临的挑战，而是问你在某个具体项目上所面临的挑战。

12.4.1　领导力和影响力

领导和影响别人可能是产品经理最基本的职责，所以这些问题出现频率如此之高也就不足为奇了。产品经理跟人事经理不一样，经常要在没有直接权力的情况下影响他人，所以面试官要知道你用来构建团队、说服或影响别人的策略。

一些常见的策略包括：

- 收集数据支持你的结论；
- 理解和询问员工的潜在动机或诱因；

- 先争取团队关键成员的支持，然后以此为基础争取其他人的支持；
- 展示自己的不足，以鼓励其他人展示自己的不足；
- 以身作则；
- 先在一个共通的框架上达成共识，然后慢慢将人们引向一个结论；
- 树立威信，培养信任。

如果这些都没能唤醒你的记忆，回想一下你影响别人时的情境，不管是同事、朋友，还是家人。你用的什么策略管用了？或者对方用什么策略影响了你？

问题示例

(1) 讲一个你曾经作过的，不受欢迎的决定。 你是怎么实现它的？
(2) 讲一个你激励员工或同事的经历。
(3) 讲讲你展示主动性的经历。
(4) 讲讲你给持反对意见的同事做演示的经历。
(5) 讲讲你必须作一个不受欢迎的决定的经历。
(6) 讲讲你向其他人或团队推销你的想法的经历。
(7) 讲讲你构建团队的经历。

12.4.2 挑战

关于挑战的问题实际上并不太关注是什么样的挑战（尽管挑战应该是有意义的事情），更多是要看你面对挑战时的反应，毕竟工作肯定要面对挑战，他们必定想知道你如何解决问题。

要对你曾经面对的挑战或问题进行头脑风暴，想想你在工作中是否碰到过下面这些问题：

- 道德困境；
- 相互矛盾的激励措施；
- 资源不足（时间、金钱、专业人员）；
- 不完整或不准确的信息；
- 士气低落，人际关系问题，或其他情绪上的问题（跟队友或整个团队）；
- 文化或工作方式上的冲突；
- 变化的需求；
- 不能完成任务或达到期望。

因为你要面对范围如此广泛的问题，所以这里也要有很多解决问题的办法。一些常见的策略有：

- 收集信息决定该做什么；

- 利用身边人的支持和专业素养；
- 讨论并设定团队的优先级；
- 理解身边人的情绪；
- 考虑什么是“正确的”事（基于客户利益最大化之类的衡量标准）；
- 分解状况，专注于你知道的，并且更深入地了解你知道什么；
- 降低风险；
- 要诚实并且简单直接；
- 创造性地解决问题，或者说思考时不受条条框框的限制；
- 折衷；
- 平衡短期和长期的利益权衡；
- 管理同事和客户的期望。

其中有些挑战可能源自工作之外的活动，如果你曾经在这里做过一些实质性的工作。然而，你同样应该准备正式工作中发生的故事。

问题示例

(1) 讲一个你曾经面对挑战并最终克服它的经历。
(2) 讲一个你没能赶在截止日期前完成任务的经历。
(3) 讲一讲你负责的工作中发生的重大改变，你是如何适应那个改变的？
(4) 讲一个你必须应对变化的优先级的经历，你是如何处理的？
(5) 讲一个你必须快速做出决定，或者没有足够数据的情况下做出决定的经历。
(6) 讲一个你在短期内使用了大量数据的经历。
(7) 讲一个你处理危机状况时的经历。

12.4.3 错误与失败

你的面试官中至少有一个会问及失败或错误方面的问题。回答这个问题时，你的目标跟回答其他行为问题一样，是要推销你自己。但这个问题中的“推销自己”的方式不同于以往其他问题。

你的面试官想要寻找以下要点。

- **一个大错误** 面试官想要看到你之前真正的失败。不管这种想法是对还是错，但很多人相信如果你没有失败过，那说明你还没有真正尝试过。人不可能一直都那么完美。因此，和你的直觉不一样，你需要有实质性的失败经历。
- **谦逊** 面试官还想看看你能否承认失败。他们知道你曾经失败过，但你敢于承认吗，特别是在一个敏感的时刻（比如面试时）？或者你要尽量掩饰它？在这个问题上你一定要诚恳。

- **处理** 面试官想看看你是如何处理这种状况的。你会纠正那次错误吗，或者会采取一些措施防止错误再次发生吗？你如何将这些信息传递给你的管理者以及/或者团队？你从中学到了什么？

要找到这样的好故事，一个办法是想想你在自己职业生涯中学到的东西；错误经常表明你学到了什么。比如说，你可能完全按照客户的要求做了产品，但他们根本不用它。这教会你要更深入地挖掘客户的动机。这样的故事很棒，因为它们很自然地带出正面的结局：你学到了一些重要的东西，这使得你变成了更好的产品经理。

有少数应聘者做得太过了，他们的错误碰到了“红线”。如果你的答案中有严重错误并/或者真正伤害到某人的事情，你应该找些别的事情讲。错误有个“最佳位置”，大多数应聘者在“太弱”这边，但有一小部分到了“哇！太过了”那边。

特别是不要让你的答案妨害你的诚实或诚信。在分析数据或试图承担太多职责时犯错误都没问题，但如果你撒谎或作弊了，那就另当别论了。

问题示例

(1) 讲一个你曾经犯过的错误。
(2) 讲一个你曾经失败的经历。
(3) 讲一个你曾经错误分析了状况的经历。
(4) 讲一个你对自己感到失望的经历。
(5) 讲一个你不能兼顾自己所有职责的经历。

12.4.4 成功

回答成功问题时是你让自己闪光的时间。回答这个问题时，你的答案可能会涵盖其他问题，不过没关系。你的成功可能是你征服过的巨大挑战，带领团队取得成功的时刻，甚至是克服了潜在的错误或失败的状况。

要回答这些问题，首先想想你觉得最引以为豪的事情。你为什么对这次行动感到自豪？是因为它困难吗？是因为它对公司有巨大的影响吗？因为它在你的舒适区之外？所有这些都可以成为很好的答案。挑一些对你有意义的事情。

一定要能跟面试官解释清楚为什么你把这个经历看作成功。如果是因为它对公司有重大影响，应该量化其影响。如果它在你的舒适区之外，则应该能以某种小方式改变你，或帮你认识自己的某些方面。如果是解决了一个困难的问题，你应该学到了一些东西。

问题示例

(1) 讲一些你觉得自豪的成就。
(2) 讲一个你达成重要目标的经历。
(3) 讲一个你在工作之外取得的特定见解。
(4) 讲一个你超出职责要求的经历。
(5) 讲一个你在问题变得严重之前解决掉它的经历。
(6) 讲一个你展示出创新性的经历。
(7) 讲一个你用创造性方式解决问题的经历。

12.4.5　团队协作

团队协作问题是用来评估你的人际关系技能的，特别是你跟同行直接合作时。在你的工作经历中找一找涉及人际沟通的时刻，或工作风格上的差异影响你的团队动态的时刻。

你会如何解决那些问题？可能的方法包括：

- 作出惊人的让步；
- 想办法让团队成员感到被重视；
- 能够赞同对你不利，但有益于团队的事情；
- 理解他人的潜在动机和诱因；
- 激励团队和鼓舞士气；
- 放弃自我，并鼓励其他人也效仿你；
- 设置共同的目标、指标和程序；
- 平衡自主权和团队凝聚力；
- 为身边的人建立自信心；
- 赋予个人更多的责任；
- 以身作则；
- 承担个人的职责；
- 展示对同事的同情和理解；
- 明确并划分职责；
- 分享知识和职责；
- 消除消极的队友或状况引起的破坏；
- 在整个团队中建立相互信任的氛围。

培养积极的团队动态没有“正确”的方式。回答这个问题时，优秀的答案应该能体现出好的团队协作要取决于具体的状况。

问题示例

(1) 讲一个你要做整个团队的工作来取得某些成就的经历。
(2) 讲一个你在工作中遇到分歧的经历。
(3) 讲一个你必须做一些自己不愿意做的事情的经历。
(4) 讲一个你必须妥协的经历。
(5) 讲一个你必须解决冲突的经历。
(6) 讲一个你跟同事之间有挑战性互动的经历。

第 13 章 估算问题

我可以和你打赌：没有人会在乎你是否知道曼哈顿一年能卖出多少份比萨。即便你知道“正确答案”，也没什么用。实际上，所谓的“正确答案”可能还会影响你的判断。也就是说：正确答案并非最好的答案。

估算问题，考察的完全是你解决这些问题的过程。这就好比旅行中重要的是旅程，而不是目的地。面试官会利用这些问题来考察你分析、解决问题的能力。

可能你也有疑问：这和做产品经理有什么关联？其实，这关联还挺重要的。

显然，一名理想的产品经理应该善于解决问题并擅长数学。此外，懂得估算其实也是一种宝贵的技能。毕竟，在现实中，即在面试中遇到这种问题，你只要力争得到一个接近的数字即可，而无需太过精确。

13.1 解题方法

既然估算问题本质上就是解决复杂问题，那么毫无悬念，它的关键就在于解决难题的方法。为了实现这个目标，我们可以将这些问题拆解成一些具体步骤（以下还有一些提示和技巧）。

步骤 1：明确问题

首先，如果不了解问题的症结，就无从下手。因此必须确保听懂了问题并且清楚对方提问的要点，这一点非常重要。

举个例子，如果面试官问，每年 Gmail 在广告上赚了多少钱，你可以先重复并确定这个问题。如果交谈的语言不是母语，或者你觉得可能听错了问题，这一点就尤其重要。

接着，可以询问一些不明确的事情。就以上问题为例，这个问题本身就有不少含糊的地方，比如说：

- 赚了多少钱，是指利润还是总收入？
- 如果是指利润，那么成本包括什么？员工薪酬计入成本吗？Gmail 的兼职员工薪酬应如何计算？
- Gmail 指的是 Gmail.com 吗？有些公司的域名使用 Gmail 主机，是否也包括在其中？
- 您问的是去年吗？还是自 Gmail 发布以来这些年的平均值？
- 广告的范围是美国，还是全世界？

每一个问题的答案都会在实质上影响你解答问题的方法和结果。

步骤 2：把你知道的（或想知道的）列出来

一旦清楚了问题是什么，就能对你所掌握的知识和需要推算的内容有一个大概的构思。在一些面试中，你可以向面试官寻求一些关键的材料。

通常，由于解决问题的并不拘泥于一种方式，你自身所掌握的知识和信息就对你可能采用的方法有引导作用。

举个例子，如果你正在推算 Gmail 在美国每一年的总收入，你可能掌握（或能大致猜到）的知识就诸如以下：

- 美国人口数量；
- 美国拥有电脑的人数占美国总人口的比例；
- 美国的失业率；
- 谷歌的年度总收入；
- 一般广告的点击成本；
- 广告的点击率；
- 和搜索式广告相比，Gmail 或其他嵌入式广告是否赚钱；
- Gmail 上的广告数量。

可能你可以向面试官询问其中一些情况，但是要作好准备，有些数据还需要你自己推算。如果这些问题中有其他候选人都知道而你不知道的（如美国人口数量），那么为了公平起见，还是问吧。

如果有疑问，不妨问一个开放性问题：“您能告诉我广告的点击率吗？或者您更希望我来推算这个数字？”如果这个面试官把这个皮球推回给你：“你觉得点击率是多少？”这可能就意味着，你太依赖这些问题了。

步骤 2 与步骤 3 是并行的。在构思你的方程时，可以回到步骤 2 的列表来看看。

步骤 3：构思方程

这可能是整个估算过程中最重要的部分了。接下来要讲的很重要，成败就在此一步。

你需要列出解决这个难题的方程。这样做能加快你解决这个问题的过程，还能让面试官看出，你有迎难而上的能力。

任何难题都有不同的解决方法，但这当中有一些方法会更高明一些。然而，基于你所掌握的知识以及背景，一般并无“最佳方法”。

假如你需要计算 Gmail 在美国一年的总收入，那么下面这个方法可能对你有帮助：

```
[# Gmail 在美国的用户数] × [用户年均广告点击数] × [平均每次点击广告的收入额]
```

如果你掌握了以上这些值，那么将它们相乘便能得到总收入了。

但这并不是唯一的方法，你也可以试试这样写：

```
[谷歌美国总收入] × [广告收入占谷歌收入的比例] × [嵌入式广告（非搜索式广告）收入占所有广告收入的比例] × [% Gmail 嵌入式广告收入占所有嵌入式广告收入的比例]
```

不要只研究某个方程，不妨来个头脑风暴，多写几个方程，然后好好计划如何推导出每一个方程的项——想必你也不想在一个项上花费了大把时间之后，突然发现根本无法推算下一个项吧。

步骤 4：考虑边缘情况和备用源

先别急着解决，停下来想一想你的方法会有什么潜在障碍。这可是个好机会，趁机向面试官展示你专注导向、勇于挑战自己的一面吧。没有人会愿意雇佣一位只会将问题束之高阁的员工哩！

举个例子，如果你要计算美国有多少个比萨店，你是否考虑了大学城？或者说，如果你要计算清洗你所在城市的所有窗户需要花费多少钱，你是否考虑了那些破了的玻璃（我估计你不会清洗这些玻璃），是否考虑了汽车和巴士的玻璃？

是否还有一些资源是你没有考虑入内的？比如说，如果你要计算每年售出的枪支数量，你是否考虑了黑市买卖，是否考虑将卖给警察的枪支也计入，是否考虑了田径赛场上使用的发令枪？

仔细考虑，看看还有哪些情况无法与你在上一步骤中创建的思路互相吻合。

有些候选人会感到忐忑不安，不敢在面试官面前揭这个问题的短。别怕！面试官大概已经问过几十遍了，不管你提不提出来，他对这个问题的缺陷都一清二楚。

步骤 5：拆解方程

此时，相信你已经有了一个成熟的思路可以拿下这个难题了。你有了一个能得到答案的方程，

甚至也已经分析了潜在的特殊情况和问题。

现在你得想办法推算这个方程中各项值。假设我们正在推算 Gmail 在美国的年度总收入，并且选定了如下方程：

```
[# Gmail 美国用户数] × [平均每个用户一年的广告点击数] × [平均每次点击广告的收入额]
```

要如何算出美国的 Gmail 用户数呢？如果面试官不提供这个信息，我们可以用这样一个方程推算（没错，又一个方程）：

```
[# Gmail 美国用户数] =
      美国人口总数
    × 使用电邮的人口数量占总人口比例
    × 使用 Gmail 的人口数量占使用电邮的人口数量比例
```

也许你恰好最近在新闻里听到过 Hotmail 用户数量。要是这样的话，正好可以派上用场了，你可以这样考虑：

```
[# Gmail 美国用户数] =
    # Hotmail 用户数量 × Gmail 和 Hotmail 的用户数量比
```

甚至这样一个方程也行：

```
[# Gmail 美国用户数] =
    # 智能手机用户数 × [
    % Android 手机市场份额比 × % 使用 Gmail 的 Android 手机用户占 Android 手机份额比例 +
    % 苹果手机市场份额比 × % 使用 Gmail 的苹果手机用户占苹果手机份额比例 +
    % 黑莓手机市场份额比 × % 使用 Gmail 的黑莓手机用户占黑莓手机份额比例 +
    % Windows Phone 市场份额比 × % 使用 Gmail 的 Windows Phone 手机用户占 Windows Phone 手机份额比例
]
```

最后一个方程看起来可能有点夸张，但事实上它也是合乎逻辑的：Gmail 的使用率可能和它的收入有关，智能手机使用率同样如此。

给主方程的每一个项都新建一个子方程，使得主方程每一个组成部分互相独立，不要把它们都揉进一个大型方程里。你还可以在纸上（或白板上）画线，将这些计算过程都划分开来。

步骤 6：回顾及声明你的假设

这时，我们已经有了一系列方程，剩下的就是猜测一些数字了。这可能得依赖于你的感觉和以前的人生经历。平时看到 Gmail 电邮地址和其他电邮地址有多频繁，对此，应该多少有点印象吧。

要注意你的主观性。没错，你认识的人都使用智能手机，但这是否代表全美国的人都是一样？那可不一定哦。

选一个简洁的整数，并向面试官声明你的假设。持续更新步骤二列出的清单，或者另想办法清楚标记下你做假设的地方。

确保向面试官说明作这些假设的原因，你的推论和思路比给出精准的数字更重要。

步骤 7：计算

最后剩下的就是将这些数值相乘。记住，我们只是要给出一个大概数字，并不是绝对精准的数字。你可以为各项设定一些简洁的整数，使方程更方便计算。

步骤 8：合理性复查

呼！终于得到答案了。把答案交给面试官，但是在此之前，一定马上要将整个流程复查一遍。确保所有的数值都是合理的。

举个例子，假设你最终推算出 Gmail 在美国年度总收入为 50 亿美元。这听起来合理吗？这意味着 Gmail 平均在每个美国人身上赚了 16 美元。要知道，美国大部分人都不是使用 Gmail 的，这个数字听起来颇令人震惊！

此时就应返回检查整个计算过程。如果确实有问题，那大概会在以下这些地方出现：

- 主方程
- 某些假设
- 算术

这个时候，考虑问题本身的合理性也大有裨益。如果你发现 Gmail 在美国有 1 亿 5 千万用户，那就是个问题。我们相信，有一半的美国人不使用 Gmail，这才合乎常理。

13.2 数字小抄

由于这类问题考察的重点并不是答案的对错，一些有用知识不妨记下来。

如果你有需要知道的事情，可以询问面试官。但是在很多情况下，可能他们还是希望你自己推算这些值。

近似值	数据内容
3亿	美国人口总数
3	平均每个家庭的人数（美国）
1亿	美国家庭总数
80岁	平均预期寿命（美国）
65至70岁	平均寿命（世界）
70亿	世界人口总数

（续）

近似值	数据内容
7亿	欧洲人口总数
40亿	亚洲人口总数
9000	一年小时数
500 000	一年分钟数
---	公司总收入额
---	公司盈利额
---	用户数

注意，表中的数值都是简洁的整数，这是为了让它们更便于计算。

在面试之前，尽量先调查一下这家公司的收入额、盈利额和用户数。当然，对一些创业公司来说，有些数字（也可能是全部数字）都可能是 0。

13.3 窍门与技巧

估算问题需要应用到大量数学知识。不管你的心算能力如何，以下的提示与技巧都能让你更轻松一点。

13.3.1 小窍门：整数

很多时候，做为一个完美主义者，注重细节都是非常可靠的品质，但处理估算问题不要这么做。毕竟，有那么多数字是你瞎猜和估计出来的，数字有些偏差真的不会有什么影响。只需要得到一个大概的数字，绝对精确的估算不会为你带来任何优势。

范例：

- 美国人口总数是 3 亿 1 千 4 百万（2012 年），应以 3 亿来计算；
- 一年有 8760 个小时（闰年有 8784 小时），应以 9000 小时或 10 000 小时来计算。

13.3.2 技巧：“72”规则

这是一个好玩又有用的小窍门：如果你需要计算一个事物翻倍增长的时间，那么用 72 除以它年增长率的 100 倍，就是其所需的时间。

也就是说，一项投资、人口总数、薪酬或其他每年增长 x%的数值，过上个 $72/x$ 年就会翻倍。

这条规则在一些 x 值较小的问题中相当实用（在正确答案 ± 5%到 ± 10%的范围内）。不过，哪怕是那些年增长率达到 100%的数值，得出的结果也在实际答案的 ± 30%范围内。

- ❑ $x < 20$：结果在实际答案的 ± 5%范围内。
- ❑ $x < 65$：结果在实际答案的 ± 20%范围内。
- ❑ $x < 100$：结果在实际答案的 ± 30%范围内。
- ❑ $x > 100$：结果会愈加不准确，但这也意味着这个数值在一年内就能翻倍。

当然，如果没有计算器，就要拿个数字来除 72，一般很费劲。相比之下，使用 70 或 75 就好算多了，得出的结果与实际答案也比较相近。

范例：

1. 薪酬增长问题

- ❑ 面试官："一个应届大学毕业生的年薪是 65 000 美元。如果她每年的薪酬涨幅是 9%，那么多久之后，她的薪酬比最初的薪酬多一倍？"
- ❑ 候选人："这个问题可以使用'72 规则'，72 除以 9 得到 8，所以需要 8 年的时间。"
- ❑ 答案：准确答案是再过 8.05 年，她的薪酬会翻倍。而 8 年之后，她的薪酬涨幅是 99%。这个四舍五入做得挺漂亮！

2. 人口增长问题

- ❑ 面试官："据 2012 年美国人口普查得出的结果，美国人口总数大约是 3 亿，年增长率是 0.7%。假设增长率保持稳定，多久之后，美国人口会达到 6 亿？"
- ❑ 候选人："这个问题的核心就是美国人口总数翻倍所需要的时间，所以我们可以使用'72 规则'，用 72 除以 0.7。那么结果是多少？结果肯定是在 72 到 144 之间（因为 0.7 就在 1 与 0.5 之间）。所以就可以猜测是在 100 年左右，也就是说，人口总数大约在 2112 年会翻倍。
- ❑ 答案：尽管我们使用了极为"宽松"的算术，候选人还是得到了一个相近的答案。如果用 72 除以 0.7，得出结果是 103 年，而实际答案是 99.3 年。候选人的答案和这两个答案恰如其分地近乎一致，但更重要的是因为他选用了一个合乎逻辑的定量方法。这就是这类问题要考验的关键所在！

13.3.3 技巧：数量级

两个大数字相乘时，很容易出岔子。如果是个位数倒没什么，但要是多了一个 0 或少了一个 0，结果可就是 10 倍甚至更多了。那你就惹上大麻烦了。

因此，最好能够确保你的答案和预期的答案是一样的数量级。有一个实用的技巧：假设 a 乘以 b 得到 n，那么 n 的位数应不大于 a 与 b 的位数之和。

或者更精确地说：

要么是：a 的位数+b 的位数=（a×b）的位数

要么是：a 的位数+b 的位数=（a×b）的位数+1

举个例子，823×1032 的乘积数要么有 6 位数，要么有 7 位数（实际上，它有 6 位数）。

所以，如果你最终得到类似 84 936 这样的数字，就该知道一定是某个环节出了错。

13.3.4 小窍门：自信一点

有多少人对你说过诸如“我的数学糟透了”“我一看到数字就犯晕”此类的话？不管是不是实话，面试可不是讲这些话的场合。

一般来说，有自信心的产品经理才是公司想要雇佣的——或者至少是假装有自信心的产品经理。如果你被问题吓住了，千万不要表露出来，更不要告诉面试官你不善于数学。自嘲的话在这儿可帮不上什么忙。

13.3.5 小窍门：给数值标上单位

很多人都把给数字标单位（如“4 米”）这事看作是只有学校老师才会坚持的烦人事情之一，但这个习惯在这里是大有裨益的。

候选人常犯的一个错误，就是被那些数字的单位给搞糊涂了。他们写下“4”代表“4 米”，到了计算的时候，却忘了它是“4 米”而不是“4 千米”。更麻烦的是，这很不幸地成为了一个非常难发现的 bug。

如果你给所有的数值都标上了单位，从长远来看，你会得到更精准的答案。

13.3.6 小窍门：考虑多种资源

不妨假设你被要求推算出每年薯片的销量。首先你需要确保自己清楚不同的销售渠道(即“资源”)。薯片通常在以下地方有售：

- 商店，直接给顾客；
- 自动贩卖机，直接卖给顾客；
- 经销商，卖给学校、医院、电影院，等等；
- 还有许多其他地方！

多种“资源”在关于市场规模和收入的问题中十分常见，但也有其他问题会出现这个概念。

举个例子，假设你被要求推算一座长宽都为一个街区、20 层高的大厦需要电梯的数量，你

13

大概会一边计算着大厦的常出入人数和他们需要搭电梯的频率，一边呼哧着累得够呛。那么，你考虑过货梯没有？

这也是一种“资源”！

很多情况下，你可以直接忽略一些补充资源，这倒没多大关系，但事先最好还是向面试官声明。这能体现你对细节的重视。

13.3.7 小窍门：保持各项计算的独立性

尽你所能，能多有条理就多有条理，越有条理越好。如果你遇到的估算问题有若干相互独立的演算步骤，那么你应尽量分别推算它们。如果过程中出了错（或者不说“如果”，这几率可不低！），你就能回头找到那个步骤，快速缩小出错的范围至准确的位置，费最小的劲儿把它改正过来。

范例：

假设你要推算 Gmail 广告上的年收入额，你的方法可能是这样的：

```
$ = [# Gmail 用户数] × [# 用户年均广告点击数] × [每次点击广告的收入额]
```

要解决这个问题，你应当将这几个项独立开来计算。你可以使用这样一个表格：

每次点击广告的收入额	用户年均广告点击数	Gmail用户数
…	…	…

在问题的最后一个步骤，你才需要合并这些数值（或者说相乘）。

13.3.8 小窍门：记录演算的过程

有些估算问题比较简单，但多数还是相当繁复冗长，需要大量计算。边演算边记下演算的过程是很重要的，因为稍后你可能需要回来给自己纠错，或是使用一个先前得出的数字。

这份记录要有条理、易读，方便稍后可以返回查看任一步骤。就当作是给数学老师演示你的计算过程吧。如果你的方法足够有条理，其他人也能理解当中思路的话，要分析哪儿出了岔子，应该就容易得多。

13.3.9 小窍门：记录你的假设

要是推算出的最终答案太离谱，也用不着太惊讶。这种情况一般由两个原因造成：要么你在计算上出了错（这种情况下，条理性就能帮你快速定位问题），要么你的其中一个假设不对。

因此，越是容易找到假设的地方，就越容易发现潜在问题。

比较理想的做法，就是在纸或白板的边上列一个清单，列下你所作的全部假设。不过要是你不习惯这样做，至少也要在推算过程中，将每个假设圈起来，便于后续查漏补缺。

13.4 面试示例

在自己进行一些估算问题练习之前，先了解该类面试的大致流程也很有帮助。下面我们将提供一段模拟对话。为了方便阅读，我们在对话中标了段落标题，也使读者更易于区分候选人所在的环节。

面试官：我们就从一些估算问题开始吧。以你的感觉，美国的洗发露行业一年能赚多少钱？

13.4.1 明确问题

- 候选人：嗯，你想要了解洗发露行业每年能赚多少钱，对吧？请问你说的是盈利还是总收入呢？
- 面试官：那就总收入吧。
- 候选人：非常好。那么是只考虑洗发露的销售呢，还是要将护发素和其他相关产品一起计入呢？
- 面试官：只考虑洗发露就好了。
- 候选人：好的。我们说要推算“收入”，我猜想应该是指洗发露公司的收入，对吧？也就是说，我们并不计入分销商的收入，对吗？
- 面试官：没错。
- 候选人：好的，非常好。我想我已经得到了需要的信息，可以开始计算了。我想确认一下，我们要推算的是洗发露销售一年的总收入，不包括相关产品。我能花点时间记下一些思路吗？
- 面试官：当然可以。

13.4.2 把你知道的（或想知道的）列出来

候选人记下如下数据：

- 美国人口总数是 3 亿；
- 平均寿命是 80 岁；
- 人均一次洗头所使用洗发露的量；
- 平均一个产品在商店的溢价占它的售价的百分比。

13.4.3 构思方程

候选人写出了一个可能的方程：

[# 美国人口总数] × [# 人均一年使用洗发露瓶数] × [每卖出一瓶洗发露的收入额]

- 候选人：那么，其中一个方法就是计算特定的一个个体一年使用洗发露的瓶数，然后乘以平均每卖出一瓶洗发露的收入额。这样就能得出人均一年的洗发露销售额。这样一来，我们就能将这个数字和美国人口相乘，得出答案。
- 面试官：没错，非常好。
- 候选人：我认为这是相当不错的一个方法，但是我们还得对市场进行细分，将这个方法拆分来推算。洗发露的使用量并不是人人相同，性别、年龄不同，使用量都有区别。举个例子，女性花费在洗发露上的钱比男性更多，因为她们的头发比男性长，而且她们更常购买较昂贵的洗发露。因此，我们的计算方法要调整一下，将男性和女性独立开来，分别计算。

针对女性，我们使用如下表格：

美国女性人口总数	女性年均使用洗发露瓶数	每卖出一瓶洗发露的收入额
50% × 美国人口总数	年均洗头次数 每瓶洗发露可用次数	平均每瓶女士洗发露售价 ×（1–商店溢价比例）

针对男性，我们计算起来就比较简单了，只要将女性的计算结果按一个常量比例来减少就可以了：

美国男性人口总数	男性年均使用洗发露瓶数	每卖出一瓶洗发露的收入额
50% × 美国人口总数	女性人均一年使用洗发露瓶数 ×（男性使用洗发露瓶数/女性使用洗发露瓶数）	平均每瓶男士洗发露售价 ×（1–商店溢价比例）

我们还可以将这个方程拆分为“奢华的洗发露”和“普通的洗发露”使用者，但我觉得按男女来区分已经足够接近实际答案了。

13.4.4 回顾及声明你的假设

- 候选人：我每次洗头使用一茶匙的洗发露，一瓶标准容量为 340 克的洗发露应该能分成 100 茶匙左右。

针对女性的方程，第二项就这样解决了。

我认为，一瓶女士洗发露的售价大概是 5 美元左右，男士洗发露大概也差不多这个价格。我

相信商店的售价会比进货价高一倍，所以我就直接用了这个数字。

候选人将上面这些大胆猜想的部分加入了假设清单。

- 平均一次洗头的洗发露用量：1 茶匙。
- 男性与女性的洗发露用量比率：50%。
- 平均一瓶洗发露容量：340 克。
- 平均每 340 克可装茶匙量：100 茶匙。
- 平均一瓶洗发露（男性或女性）单价：5 美元。
- 商店溢价：50%。

13.4.5 计算

- 候选人：现在，剩下的基本就是把数值代入方程进行计算了。

针对女性用户，我们这样计算：

美国女性人口总数	女性年均使用洗发露瓶数	每卖出一瓶洗发露的收入额
50% × 美国人口总数	年均洗头次数 每瓶洗发露可用次数	平均每瓶女士洗发露售价 ×（1–商店溢价比例）
50% × 3亿人口	每年洗头365次 / 每瓶洗发露可用100次	每瓶售价5美元 × 50%
1.5亿	3.5瓶/人每年	2.5美元/瓶

洗发露公司平均从每位女性手上赚到的钱大约是每年 9 美元，因此这一部分的结果就是 13.5 亿美元。

男性的头发比较短，他们使用的洗发露就应该更少。我们不妨这样说，男性使用洗发露的分量是女性的一半。这就意味着男性每年给洗发露的收入贡献了大约 6.5 亿美元。

所以，总的加起来，单是洗发露一种产品，他们一共收入了大约 20 亿美元。

这个数字听起来并不离谱，不过我还是要再次确认一遍。

- 面试官：好的，请便。

13.4.6 合理性复查

- 候选人：嗯，我觉得这儿可能有一个问题。针对女性，我使用了每年 3.5 瓶、每瓶 340 克的数据。也就是说，一名女性月均只使用了 100 克洗发露。这个数字听起来有点小。就拿我自己来说，在旅行中，一瓶 100 克的旅行装洗发露只够我用上大约半个月。我想，

我推算出来的洗发露使用量，不论男女，都只是事实的一半。如果我将方程中的使用量翻倍，那总收入就翻了一倍。因此，年收入额应该接近 40 亿美元。

- 面试官：有意思。好的，还有其他补充吗？
- 候选人：细想一下，我觉得我先前应该是作了一个不恰当的假设。我假设每人的洗发露使用量和一个典型的成人（分别为男性或女性）相符，但对于这一点其实我并不敢肯定。我并没有考虑低龄儿童、长者、残障人士以及秃顶者。就此而言，也即是说我并未考虑那些并不每天洗头的人群。我们需要就此调整一下洗发露的使用量。10 岁以下的儿童、长者以及残障人士占美国人口总数的 20%。由于头发短或洗头次数少，他们使用的洗发露量比普通成人少得多；不妨假设使用量为 0。这就将洗发露使用量减少了 20%。比方说，美国有 20%的成人男性秃顶，这就又将洗发露使用量减少了 10%。

  那么，还有那些并不每天洗头的人群呢？我们就假设有 20%的人并不每天洗头，而是两天洗一次头吧。这又减少了 10%的使用量。

  把这几个数字加起来，我们得把洗发露使用量减掉 40%。所以，40 亿美元并不准确，最终的年收入应稍高于 20 亿美元。
- 面试官：很棒！干得好！

13.5 样题

以下是 10 道估算问题，这些问题与平常面试中会遇到的问题异曲同工。我们也提供了解决方案，但仅供参考，因为每个问题都有很多正确的解决方案。尽管如此，通读这些方案（当然，建议读者先自行尝试解答）也很有用，能帮助你探索问题的替代方案，或者如果你碰巧使用了同样的方案，它们还能为你提供细节上的参考。

这些问题大多描述得含糊不清，但这没关系！作为候选人，你的目标之一就是在解决问题之前，先消除这些含糊之处。

要开始计算这些问题，首先你得作一些适当的假设，来消除题中那些模棱两可的表意。在给出的解决方案中，我们可能会使用一些与你所作不同的假设。这没关系，我们的方案并不等同于正确答案，只不过是一种合理的方法。

在可能的情况下，我们从某些调查统计或其他数据获取了“正确答案”。我们提供这些数字只是为了好玩，所以如果你的答案（或我们的答案）与正确答案相距甚远，不必过分担心。弄明白自己的答案为什么会有那么大的偏差，是很有帮助的——是逻辑上犯了错误，还是作了不合理的假设？而实际答案不管有多精准，都没有什么大不了。

我们为前三个问题逐步列出了详细的解决方法，以便你掌握其中的窍门。

样题 1：美国人每年花费多少钱在狗粮上？

现在，让我们逐步分析这个问题。

步骤 1：明确问题

这个问题有哪些地方是含糊不清的？想想看。

- 湿粮与干粮是否一并计入？（假设是。）
- 这里指的是终端消费者还是商店？（假设是消费者。）

这个问题并无太多歧义，所以我们可以带着这两个假设继续往下走。

步骤 2：把你知道的（或你想知道的）列出来

假设目前我们知道或想知道以下信息：

- 美国人口总数是 3 亿；
- 平均每个美国家庭有 3 个人；
- 有多少人养狗？
- 大型犬只每天进食 1 至 2 次，大致能吃完一个标准大小的犬用食盆的量；
- 一袋中等大小的狗粮大约重 9 公斤；
- 一袋狗粮的价格是多少钱？
- 大部分的狗都吃干粮；
- 平均一位狗主人养多少条狗？

步骤 3：写等式

如下等式应该就能推算出答案：

```
[# 美国犬只数量] × [犬只平均一年的进食量] × [狗粮单价]
```

此时应停下来，仔细考虑这个方法的可行性。等式的每一个项，我们是否都能推算出来？是否还有其他方法可以使用？

步骤 4：考虑边缘情况和备用选项

我们的方案主要专注于宠物的数量。但是，其他地方是否也会养狗？那当然！

- 警察部队
- 赛狗场
- 农庄

不过，这并不会有什么影响，因为狗主人的数量实在太大了。

步骤 5：分解等式

推算犬只平均消耗的狗粮数量以及每袋狗粮的价格，这应该是小菜一碟。但是要推算犬只的数量可就棘手多了。

我们可以大胆猜测一下美国家庭养狗的比例，但若是将美国家庭再划分开来分析，可能会更好。我们可以通过以下几种方法将美国家庭划分开，以分析犬只的数量。

- 按收入水平划分。
- 按地理位置划分：郊区、城市、农村。
- 按年龄划分：18 岁 ~ 30 岁、30 岁 ~ 60 岁，60 岁以上。18 岁以下的孩子不计入，因为他们一般是与成人一同居住。
- 按居所类型划分：公寓与独立住房。
- 按是否育有孩子划分。

以上任何分类方式（或其他方式）都可以，也可以结合使用不同的划分方法（比如说，按年龄和收入水平划分），但这样可能会使计算过程变得更复杂。

下面，我们将使用最后一个划分方法，将这个问题按有孩子的家庭和没有孩子的家庭来进行估算。

# 美国犬只数量	犬只平均一年的进食量	狗粮单价
[# 美国家庭数量] × [% 养狗的美国家庭占全部家庭的比例] × [养狗家庭平均养狗数量]	[犬只平均一天进食量] × [每年进食天数]	袋装狗粮价格 / 袋装口粮份量

```
% 养狗的美国家庭占全部家庭的比例
    = [% 有孩子的家庭占全部家庭的比例] × [% 有孩子的家庭中养狗的家庭比例]
    + [% 没有孩子的家庭占全部家庭的比例] × [% 没有孩子的家庭中养狗的家庭比例]
```

步骤 6：回顾及声明你的假设

- 假设 30%的美国家庭育有孩子。我之所以猜测这个数字，是因为某种程度上来讲，90%的人都育有孩子，而孩子需要 20 年的时间来长大。也就是说，有孩子的人，孩子在身边的时间一般占他们成年生活时长的 1/3。
- 假设 30%育有孩子的美国家庭养狗。
- 假设 10%未育有孩子的美国家庭养狗。
- 假设 20%养狗的美国家庭养了 2 只狗。也就是说，平均每个养狗的家庭养了 1.2 只狗。
- 假设养有 2 只以上的狗的家庭比例小到可以忽略不计。
- 假设普通的狗每天能吃 1.5 杯狗粮。我养的一直都是大型犬，它们每天能吃 2 杯狗粮。它们的食量大概是按体型大小比例来的吧，所以，小型犬的食量应该是每天 1 杯左右。
- 假设一袋标准大小的狗粮大约 9 公斤重，售价 15 美元左右。

❑ 我认为 9 公斤重的狗粮大约能分成 20 杯。

步骤 7：计算

现在，就剩把数字代入等式了。记住，要使各项的单位都井井有条。

```
% 养狗的美国家庭占全部家庭的比例
    = [% 有孩子的家庭占全部家庭的比例] × [% 有孩子的家庭中养狗的家庭比例] +
      [% 没有孩子的家庭占全部家庭的比例] × [% 没有孩子的家庭中养狗的家庭比例]
    = 30% × 30% + 70% × 10%
    = 9/100 + 7/100
    = 16%
```

# 美国犬只数量	犬只平均一年的进食量	狗粮单价
1亿美国家庭 × 16% × 每个养狗家庭1.2只狗	日均每只狗进食1.5杯 × 每年365天	袋装狗粮价格 / 袋装口粮分量
2亿只狗	年均每只狗大约500杯	每杯1美元

将这几个数字相乘，我们就得到结果：每年花费 100 亿美元在狗粮上。

步骤 8：合理性复查

这个数字听起来合理吗？你可以通过这些方式来检查。

❑ 年均花费 500 美元在一只狗的狗粮上，听起来合理吗？
❑ 16%的美国家庭养狗，这个数字合理吗？
❑ 如果美国有 2 千万只狗，那么平均每 15 个人就有一只狗。这听起来合理吗？

如果这个答案存在较大的问题，那么会是哪儿出了错呢？

此时，可以这么考虑。

❑ 假设/推算出 30%的美国家庭育有孩子。这个数字应该不会有太大出入（反正不会是 5%或 80%），但是实际的数字应该是 20%到 60%之间。
❑ 假设 30%育有孩子的家庭养狗。这只是一个大胆的猜想，所以这可能就和问题有关。
❑ 假设 10%未育有孩子的家庭养狗。同样，这也是一个大胆的猜想，这个数字也有可能和实际相差甚远。
❑ 假设 20%养狗的家庭养有 2 只狗。

针对潜在的问题进行头脑风暴，如有必要则修正计算过程，这在一场面试中是很有价值的。

实际答案

美国实际上有 7.82 亿只狗[①]，这和我们的答案足足相差了 3 倍。这不算糟糕，我们的答案还

① “Pets by the Numbers.” Human Society. 27 September, 2013.

是在大概数字的可接受范围内。到底是哪儿出了错呢？动物保护协会提供了一些有用的统计数据：

- 39%的家庭养有 1 只以上的狗（我们的猜测是 16%）；
- 60%的狗主人养有 1 只狗（我们的猜测是 80%）；
- 28%的狗主人养有 2 只狗；
- 12%的狗主人养有 2 只以上的狗；
- 平均每个养狗的家庭养了 1.69 只狗（我们计算出了 1.2 只狗）。

这么说，我们低估了养狗家庭的数量了（16%，而实际是 39%），导致结果出现了 2 倍偏差。我们还稍微低估了养多只狗的家庭数量。我们算出平均每个狗主人养了 1.2 只狗，而实际上是 1.69 只。

此外，我们的狗粮也出现了 2 倍的偏差。据宠物网站 PetFinder 估计，一只狗年均狗粮费用为 250 美元[1]。而我们计算出的数字是 500 美元。美国人年平均应该为狗粮花费 200 亿美元，而我们计算出的数字是 100 亿美元。

样题 2：一套二居室的公寓能装下多少个网球？

这和一道经典的谷歌面试题很像：一架波音 747 飞机能装下多少个高尔夫球？

步骤 1：明确问题

这个问题和很多问题一样， 也有一些含糊不清或未明确说明的地方，举例如下。

- 这里说的二居室公寓，是特指某一套公寓（可能是任意大小），还是随便一套典型的二居室公寓？（假设是一套典型的二居室公寓。）
- 理解一个“典型”的二居室公寓是什么意思，首先我们需要知道它的所在地。这是美国的一套二居室公寓吗？（假设是美国商业区的一套典型二居室公寓。）
- 这个问题问的是，一套典型的二居室公寓最多能装下多少网球，还是我们能随意投掷多少个网球进去？（假设问的是能装下的网球最大数量。）
- 这套二居室公寓中是否有家具？有多少家具？公寓里是否有人？（假设公寓中有一般数量的家具，但没有人。）
- 我们可以将网球塞到橱柜里吗？（假设不行。所有的网球都必须放在“目所能及”的地方。）

步骤 2：把你知道的（或你想知道的）列出来

目前我们知道的或想知道的信息，包括以下内容。

- 据我的个人经验，一套典型的二居室公寓大约是 75 平方米。

① How much does owning a pet cost in a year?” Kay, Liz F. 12 February 2012. Bankrate.com.

- 一套典型的二居室公寓通常有 2 个卧室、1 个起居室、1 个厨房和 1 个浴室。
- 家具一般包括：2 张双人床、2 个梳妆台、4 张床头柜、1 张长沙发、1 张双人沙发、1 张餐桌和 4 张餐椅、1 张咖啡桌和一台电视机。
- 厨房用具占了多少空间？浴室用具呢？

步骤 3：写等式

基本等式是非常简单直接的：

```
([公寓的空间大小] – [家具体积] – [其他用具体积] – [个人物品体积]) / [网球的单位体积]
```

步骤 4：考虑边缘情况和备用选项

我们已经列出了公寓里各种不同的家具和用具，这就解决了大半的潜在问题。还有一个边缘情况，就是公寓的主人太邋遢，把所有物品都放到公共地方，而不收进橱柜或抽屉里。

不过，既然我们使用了各种典型的值，这些边缘情况就不考虑了吧。

步骤 5：分解等式

现在，我们需要将这个等式分解开，来分析各个项。

公寓的空间大小

公寓的大小是由公寓内地板区域的大小决定的，因此墙壁或房间的数量并不影响公寓的空间大小。

```
公寓的空间大小 = 天花板高度 × 地板面积
```

家具体积

一套典型的二居室公寓中的家具包括：

- 2 张双人床（床上有一块褥垫和一张弹簧床垫）
- 4 张床头柜
- 1 张餐桌
- 4 把餐椅
- 1 张长沙发
- 1 张双人沙发
- 1 台电视机
- 1 张咖啡桌
- 2 张梳妆台

我们可以忽略那些只有一个框架的物品（比如说桌子），因为它们多半不会影响可用的空间。

这就将我们的家具单子瘦身为：4 个双人床褥垫（床架本身的体积可忽略不计）、1 张长沙发、1 张双人沙发和 2 个梳妆台。

```
家具所占空间大小 =
      4 × 中号床垫体积
    + 1 × 长沙发体积
    + 1 × 双人沙发体积
    + 2 × 梳妆台体积
```

生活用具及嵌入式家具的体积

一套典型的公寓包括这些用具及家具：

- 浴缸
- 水槽、浴室橱柜
- 马桶
- 厨房储物柜格（包括冰箱）

假设浴缸和马桶的体积可忽略不计，但是我们仍然需要计入水槽和橱柜的体积。

```
生活用具的体积 =
      水槽/橱柜的体积
    + 厨房储物柜的体积
```

个人物品的体积

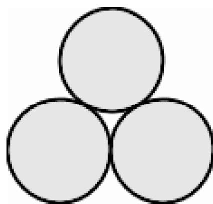
一个人最重要的个人物品十有八九就是他的衣物了。有些衣物会收在梳妆台里（这种情况不会影响到公寓装网球的可用空间），有些则收到衣柜里。

```
个人物品体积 =
    % 衣柜里的个人物品占全部物品比例
    × 所有个人物品的体积
```

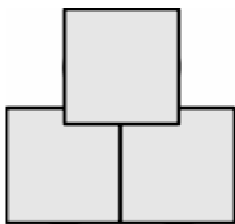
网球的单位体积

要计算网球的体积，很多人都会使用球形的体积计算方法。但这并不是很正确。除非我们将网球都磨成粉末（显然，球里面还要填充其他东西才行），否则球的体积就无关紧要了。立方体的体积才有意义，毕竟，网球之间还有缝隙。

如果网球要尽最大可能减少缝隙叠放在一起，那么它们应这样堆叠：



我们也可以大致这样表示网球的堆叠方法：



如图，我们可以看到，堆叠时令网球之间有相叠之处，这使得在指定的高度中，更多的网球得以加入堆叠。与其计算网球的单位体积，我们不如计算一个网球的“有效体积”。什么是有效体积？想象每一个网球周围有一些小方块积木，这些会占多少空间？

```
网球的有效单位体积
    = 网球长度 × 宽度 × 高度
    = 网球的直径 × 网球的直径 × 适应堆叠的高度
```

步骤 6：回顾及声明你的假设

这个问题，我们有一大堆东西要假设！我们先将假设的数值填入先前写好的等式里。

```
公寓的空间大小
    = 天花板高度 × 地板面积
    = 3.6 米 × 75 平方米
    = 270 立方米

家具体积
    =   4 × 双人床褥垫体积
      + 1 × 长沙发体积
      + 1 × 双人沙发体积
      + 2 × 梳妆台体积
    =   4 × 1.8 米 × 0.3 米 × 1.8 米
      + 1 × 1.8 米 × 1 米 × 0.6 米
      + 1 × 1.2 米 × 1 米 × 0.6 米
      + 2 × 1 米 × 0.6 米 × 1.2 米
    ≈  7.1 立方米
```

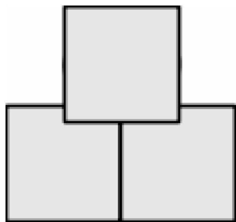
接下来是厨房，我们假设厨房大小为 3 米 × 3 米，橱柜立于地板上，在一面墙铺开，将两边的墙壁连起来，占地空间要以地板到柜台的高度来计算。

```
生活用具的体积
    =   水槽和橱柜体积
      + 厨房储物柜体积
    =   1 米 × 0.6 米 × 1.2 米
      + 2 × 3 米 × 0.6 米 × 1 米
    ≈  4.3 立方米
```

要计算个人物品的体积，我们可以回想自己搬家时的情景。搬家时，一共打包了多少箱衣物和其他用品（一般的箱子长宽高都是 0.6 米）？

```
个人物品体积
     = % 衣柜里的个人物品占全部物品比例 × 所有个人物品的体积
     ≈ 50% × 2 人 × 每人 10 箱 × 0.22 立方米
     ≈ 2.2 立方米
```

最后我们要计算网球的有效体积。我们可以通过更加严谨的算术来获得“立方球体”适应堆叠的高度值。或者干脆直接就盯着这个图片，目测出来。



```
网球的有效单位体积
     = 网球的直径 × 网球的直径 × 适应堆叠的高度
     = 6.5 厘米 × 6.5 厘米 × 5 厘米
     = 211.25 立方厘米
     ≈ 0.000211%立方米
```

仔细一点！211 立方厘米可不是 0.0211 立方米！而是 211/ (100 × 100 × 100) 立方米，或者说 0.000211 立方米。

步骤 7：计算

到了这一步，基本上就可以结束了，只需要再作一些计算。

```
公寓的空间大小
     = 3.6 米 × 75 平方米
     = 270 立方米

家具体积 = 7.1 立方米

生活用具的体积 = 4.3 立方米

个人物品体积 = 2.2 立方米

网球的有效单位体积 = 0.000211 立方米
```

我们的最终答案是：

```
公寓的可用空间
     =     公寓的空间大小
        - 家具体积
        - 生活用具的体积
        - 个人物品体积
     = 270 立方米 – 7.1 立方米 – 4.3 立方米 – 2.2 立方米
     = 256.4 立方米
```

```
# 网球的数量
    = 256.4 立方米/0.000211 立方米
    ≈ 122 万
```

步骤 8：合理性复查

122 万，听起来是否太多了？反正我是觉得有点多。我们就不管那一大堆细节了，直接对最终答案来一个快速复查吧。一套 75 平方米、高 3.6 米的公寓，大小是 270 立方米。如果这套公寓能容下 122 万个网球，那么 1 立方米应能容下 4500 个网球。

1 立方米容下 4500 个网球，这个数字听起来会不会太多？这倒不会。如果 1 米的长度能摆放 15 个网球，那么 1 立方米就能容下 4500 个网球。

计算方面检查过了，没问题。该向面试官提交最终答案了，就是 122 万个网球。

样题 3：美国有多少名警员？

这个问题有点棘手，我们马上开始吧。记住，解决方法才是重点，最终答案并不重要。

步骤 1：明确问题

这个问题最重要的歧义，就在于我们要把哪些人算作“警员”。比如说，大学校园通常就有他们自己的驻守警员。此外，文职警员是否也计入警员行列？

不妨假设这个问题指的是所有的警员，包括驻守大学校园的警员，不管他们是做文职还是巡逻。

步骤 2：把你知道的（或你想知道的）列出来

我们知道：

- 美国人口总数是 3 亿；
- 人口密度高的地区警员数量也更高；
- 每天 24 小时都有警员当值；
- 巡逻工作一般由 2 人互相搭档。

我们可能要需要知道以下这些信息。

- 美国的年均犯罪率是多少？
- 多少警员做出巡工作，多少名警员做文职工作？
- 平均每人需要多少名警员？

最后一点尤其有意思。要想大概预测所需的警员数量，可以利用一个具体的例子，比如我们知道或工作过的小镇或学校等地方。另一个要考虑的问题就是警员占美国全体雇员的比例，1%

当然是不合理的。

步骤 3：写等式

一个这样的等式大概可以解决问题：

```
([美国人口总数] / [# 人口数量/巡警数量]) × (# 警员数量/巡警数量)
```

步骤 4：考虑边缘情况和备用资源

我们已经考虑了警员的其他资源（如大学校园等），所以不用考虑什么备用源了。我们需要考虑的是犯罪率对该地区平均警员数量的影响。

步骤 5：分解等式

我们已经知道美国的人口总数。棘手之处在于如何推算平均 1 名巡警覆盖的人口数量，以及巡警在所有警员中所占的比例。我们将在下一节探讨这些问题。在这个样题中，主等式并没有需要分解的项。

步骤 6：回顾及声明你的假设

如何才能得到平均一名巡警覆盖的人口数量呢？你可能知道或假设了以下的信息。

- 我读过一篇文章，一所拥有 2000 名学生的学校有 4 名驻守警员。不过，学校还需要少数职员来看守校门。因此，校园越大，可能人均拥有警员的数量就越少；
- 一个小规模的居民小镇至少需要 6 名警员（一天轮 3 个班次，1 个班次 2 名警员）；
- 大多数警员都是男性；
- 我设想，200 个男性之中最多只有 1 名警员。那就是说，400 个人当中，大约有 1 名警察；
- 教师的数量应当比警察的数量高。每有 1 名中小学学生（6 岁至 18 岁），就至少有 1 名教师。大约 1/8 的美国人处于这个年龄区间，所以每 240 人当中就有 1 名教师为 6 至 18 岁的学生教学。

这些假设提供了一堆不同的数字，但它们都一样是大概接近的值。我们可以从中作出一些猜测，得到我们想要的数字。那我们就假设每 500 个人当中有 1 名巡警吧。

那么，我们能如何猜想，多少名警察中有 1 名巡警呢？这样想吧，你觉得是巡警多还是文职警察多？我猜想每一个案件都有许多文书工作需要跟进，所以我会假设每有 1 名巡警，就有 1 名文职警察。

步骤 7：计算

现在我们需要将假设代入到等式中：

```
([美国人口总数] / [# 人口数量/巡警数量]) × (# 警员数量/巡警数量)
```

```
=    [3 亿人口/（每 500 人/1 名巡警）]
  ×（2 名警员/1 名巡警）
= 120 万名警员
```

步骤 8：合理性复查

如何进行合理性复查呢？120 万警员意味着：

- 每 300 人当中就有 1 名警员。警员的数量比中小学教师稍小；
- 如果绝大多数警员是男性，那么每 150 名男性中就有 1 名警员；
- 纽约市大约有 26 000 名警员；
- 旧金山市大约有 2500 名警员。

除了所有男性中警员的比例之外，大部分听起来都没有什么问题。这个比例听起来有点高。

实际答案

2006 年的数据显示，一共有 861 000 名警员及警探。美国司法统计局①提供了如下事实：

- 2008 年，地方警察局共有 461 000 名警员，占所有州警员和地方警员的 60%；
- 市警察局与县警察局按照每 2.3 名全职警员服务 1000 人的比例雇佣了警员。

结果还不错！我们的答案和实际答案差不多。

样题 4：美国有多少所学校？

我们假设这个问题所说的“学校”是指从幼儿园到高中的所有学校，包括公立学校和私立学校。

针对这个问题，可以通过估算公立学校和私立学校的学生人数以及平均每所学校的规模来得到答案。

假设美国人口有 3 亿，人均寿命 80 岁，人口均匀分布在各年龄层。即是说，美国有 5000 万学龄儿童。

公立学校和私立学校的学生人数：这个问题很棘手。我们将美国人口分为低收入、中等收入、高收入三个类型。将这三个类型概念化，形成一个金字塔，其中 50%是低收入者，40%是中等收入者，10%是高收入者。

- **低收入者** 底层低收入的这一半美国人大致负担不起私立学校的费用，所以他们当中 100%（大概）的孩子上的都是公立学校，也就是说约有了 2500 万名上公立学校的学生。

① “Local Police.” Bureau of Labor Statistics. 10 November 2013.

- **中等收入** 另外这 40%的孩子当中有一小部分上的是私立学校，我们不妨当这个比例是 10%吧，相应就有了 200 万名上私立学校的学生和 1800 万名上公立学校的学生。
- **高收入** 在收入最高的金字塔顶（10%），我估计有 20%的孩子上私立学校。那么就有了 400 万名上公立学校的学生和 100 万名上私立学校的学生。

这样一来，我们就有了 4700 万名公立学校的学生和 300 万私立学校的学生了。

公立学校和私立学校的规模：要知道，公立学校和私立学校的规模是不一致的。城市里的学校规模会比较大，但很多美国人住在较小的城镇或郊区。

- **公立学校** 我们就假设一般的公立学校（小学、初中、高中）都有 4 个年级吧。规模大的公立学校每个年级可能会有 1000 名学生，但也有很多学校规模比较小（尤其是乡郊地区）。我猜想平均每个年级有 250 名学生，即平均每所学校有 1000 名学生。
- **私立学校** 私立学校很可能是初中、高中连读的，所以可能平均每所学校有 5 个年级。而且私立学校的规模一般不会变动。私立学校的一个年级可能最多 100 名到 150 名学生，最少 50 名学生吧。我们就假设平均一个年级有 75 名学生吧，那么平均每所学校就有 375 名学生。

学校数量：现在我们只需要把这些数据揉合到一起。

- **公立学校** 4700 万名学生，平均每所学校有 1000 名学生，所以一共是 47 000 所公立学校。
- **私立学校** 300 万私立学校学生，平均每所学校 375 名学生，所以大约是 8000 所私立学校。

我们得出的结果是，美国总共有 55 000 所学校，其中 15%是私立学校。

实际答案

2008 年，美国有 133 000 所学校，包括幼儿园、小学、初中和高中[①]。这些学校当中，大约有 99 000 所（75%）是公立学校。5500 万[②]在校学生，当中有 500 万学生上的是私立学校。

这并不奇怪，我们得出的在校学生数量非常准确，私立学校的学生数量也非常接近。相比较而言，学校的数量就没有那么准确了，但仍可算是相当接近。

记住，最重要的是解决估算问题的方法，而不是最终答案。解决一个这样的问题多半有赖于你的假设。哪怕你使用了一个与上述相同的解决方法，如果假设公立学校每个班级有 500 名学生，那么你的答案也会和实际答案相距甚远。不过幸好，重点是解决问题的方法，而不是最终的数字。

---

① “Digest of Education Statistics: 2010.” National Center for Education Statistics. April 2011.

② 原文是 55 000 百万（55,000 million），即 5 千 5 百亿，但根据上下文，判断为原文错误，故译本处理为 5500 万（55 million）。——译者注

样题 5：用一根吸管排光一个热水浴池里的水，需要多长时间？

对于这个问题，我会假设有个人在浴缸旁边，不断地用吸管从浴缸里吸水，然后吐掉，以此来排空浴缸（而不是把吸管当成排水管，接走一股连续不断的水流）。

要这样来估算，我们需要计算一个标准热水浴池的尺寸，一根标准吸管的容量，以及排空和装满吸管的时间。我们可以使用以下等式来计算：

([浴缸的容量] / [吸管的容量]) × [排空和装满吸管所需时间]

我们并不考虑蒸发掉的水分，也不考虑因浴缸底部的水较难排出而耽误的时间。至少在目前，我们会先忽略不计这些情况。

浴缸的容量：我们可以用“立方米”为单位来表示浴缸的容量。

- **热水浴池的深度**　热水浴池里有座位，会占一些空间。我们需要知道的是热水浴池的平均深度。想象一个身高 1.8 米的人站在浴缸里，那么浴缸里的水应该是在膝盖以上、腰部以下。那么水面到浴缸底部的深度应该是 0.8 米左右。如果这个人坐下，水就能没过他胸部的一半高。那么水面到座位的深度应该是 0.3 米左右。座位占了浴缸底部的一半面积，所以我猜想，浴缸的平均深度应是 0.6 米左右。
- **浴池壁的长度**　一个热水浴池可以同时容纳 3 个人，平均每人宽度为 0.6 米。所以热水浴池的长度大约是 1.8 米。因此，热水浴池的容量大约是 2 立方米（1.8 米 × 1.8 米 × 0.6 米）。
- **吸管的容量**　一根吸管大约 20 厘米长。我猜想它的直径应该是 0.5 厘米，即半径为 0.25 厘米。我们不妨将 π 的值四舍五入为 3 吧。还是用公制单位来计算，我们就得出了吸管的容量：20 × (3 × 0.25 × 0.25)，即大约 4 立方厘米。

用 2 立方米（即 2 000 000 立方厘米）除以 4 立方厘米，得到 500 000。所以，要使用吸管来排空一个热水浴池，需要 500 000 次来回。

排空和装满吸管所需时间：只需要亲身尝试一次就知道了，感觉大约需要 4 秒钟。这样，我们的答案就出来了：500 000 次吸放吸管 × 4 秒/次 = 2 000 000 秒。

单位转换为“天”：此时，为了更加一目了然，我们可以将单位转换成天。60 秒/分 × 60 分/小时得出每小时有 3 600 秒钟。再以 24 小时相乘（先乘以 10 得到 36 000，再乘以 2 得到大约 70 000，然后增加 25%得到大约 90 000 秒），就得出了每天大约有 90 000 秒钟了。

用 2 000 000 秒除以 90 000 秒，我们可以得到一个比 20 稍大的数字，大约是 22 天。

因此，我们的最终答案就是 22 天。

样题 6：美国每年卖出多少副眼镜？

对于这个问题，首先假设问题所说的“眼镜”仅指普通眼镜，不包括太阳眼镜。

美国人口总数是 3 亿。假设人均寿命为 80 岁，以及人们换眼镜的频率是 3 年一次。现在我们需要计算美国戴眼镜的人数，并除以 3。由于视力会随年龄增长而变坏，我们需要将人口按年龄分类。我们还需要按性别来分类，因为（根据个人经验）戴眼镜的男性数量比女性高（女性更趋向于戴隐形眼镜）。

视力问题还有近视和远视之分，这也需要独立分析。

近视：根据个人经验，儿童很少有近视问题。另外，20 岁上下、不患近视的人视力能长期保持稳定。

年龄层	近视比例	近视男性戴眼镜比例	近视女性戴眼镜比例	年龄层戴近视眼镜比例
10岁以下	0%	0%	0%	0%
10 ~ 40岁	50%	50%	20%	17.5%
40 ~ 80岁	80%	80%	80%	64%

将各年龄层占人口的比例值和表格右侧的数字一起计算，我们可以得出结论：38%的人戴近视眼镜。

远视：极少人在进入中年之前就有了远视的毛病。到了 60 岁左右，远视问题就见怪不怪了。远视的人群一般只是偶尔需要眼镜来矫正视力，所以他们更多是戴远视眼镜，而不是戴隐形眼镜。

年龄层	远视比例	远视男性戴眼镜比例	远视女性戴眼镜比例	年龄层戴远视眼镜比例
10岁以下	0%	0%	0%	0%
10 ~ 40岁	0%	0%	0%	0%
40 ~ 60岁	50%	90%	90%	45%
60 ~ 80岁	80%	90%	90%	72%

这就意味着有 30%的人戴远视眼镜。

近视或远视：美国有 3 亿人口，其中 30%戴远视眼镜，38%戴近视眼镜。也就是说，9 百万人戴远视眼镜，1 亿 1 千 4 百万人戴近视眼镜。整个美国有将近 2 亿人需要戴眼镜。（需要注意的是，有些人被重复计入了，比如说有些人既戴近视眼镜又戴远视眼镜。尽管如此，我们的目的同样能达到。）

如果每个人每隔 3 年换一副眼镜，那么每年就会卖出 6700 万副眼镜。

样题 7：一辆校巴的重量是多少？

首先，我们不妨假设问题所指的是一辆加满油的空校巴，没有乘客。同时，我们还要假设相对短途公交巴士来讲，这是一辆大型校巴。

那么一辆校巴有多大呢？据我的印象，一辆标准的校巴大约有 15 排座位。我想每两排之间有将近 1 米的距离，所以应该有十四五米长，还要再加上司机位所占的长度。那么长度应该是 16 米左右。

巴士的宽度应该比汽车大，但也不能宽得过了头，否则就不能上路行驶了。所以宽度大约是 1.8 米。

现在，我们就将巴士的零件拆分，分别来计算它们的重量。我们把巴士拆分为：座位、油箱、轮胎、车窗、引擎和车架。

座位：15 排，每边 2 个座位，一共是 30 个座位。我相信座位的材料是相当坚固的金属，所以它们不可能是超轻的。我猜想每个座位的重量是 20 多公斤，我们就说大约 23 公斤吧。那么全部座位总共就是 690 公斤左右。

油箱：我想，我的汽车油箱容量大约是 75 升，巴士的油箱容量应该大得多。就当它有汽车油箱的 3 倍那么大吧，那么它大约是 230 升。1 升水重 1 公斤，我们干脆假设汽油也是同样的密度和重量吧。那么汽油一共有 230 公斤左右。

轮胎：大型巴士可能会有 3 排轮胎，而每排有 2 个轮胎，所以一共有 6 个轮胎。我曾试过举起一个汽车轮胎，记得并不是特别重。不妨假设汽车轮胎重 10 公斤。巴士轮胎倒是大一些，可能每个轮胎 13 公斤吧。6 个轮胎，也就是说大约是 80 公斤。

车窗：我想，每扇窗户大约是 0.5 米高。我们就当车窗都是连着的，那么每一侧就有 8 平方米（16 米 × 0.5 米）的车窗面积。再计入前后的挡风玻璃面积（1.8 米 × 1.2 米+1.8 米 × 0.6 米），大约是 3.2 平方米。那么车窗总共将近 20 平方米。

那么车窗有多重呢？我也曾试过举起玻璃桌面，长宽将近 1 米的玻璃，很容易举起来，但还是显得挺重的。我猜想那大约重 20 公斤。所以我们得出，1 平方米玻璃大约重 2 公斤，也就是说 18 公斤车窗玻璃应该重大约 360 公斤。

引擎：要来猜测这部分零件的重量，有点棘手，我可不曾有举引擎的经验。然而，我会假设，一个汽车引擎需要 2 到 3 个壮汉才能举得起来。如果每人能举 50 公斤的重量，那么 3 个人就能举 150 公斤的重量。巴士的引擎应该还是比汽车引擎重得多，那我们就假设它重 230 公斤吧。

车架：车架应该是这个问题当中最难估算重量的部分了，我实在不知道金属的重量。车架下面有一个金属架，长宽大约是 14 米 × 1.8 米。它应该是相当厚实的，我猜想每 1 平方米的重量是

140 公斤左右，所以底部大约重 3600 公斤。

顶部就没有那么厚重了，那就当它是 1800 公斤吧。

巴士左右两侧的长宽大约是 14 米 × 2 米。如果我们假设每平方米重 70 公斤，那么每一侧大约重 1800 公斤。 巴士前后两侧的长度大约是 1.8 米 × 1.8 米。我猜想这两部分还是需要坚固的金属，那么，假设每平方米重 140 公斤，每一侧就是 450 公斤左右，即两侧总共 900 公斤左右。

那么，整个车架的重量就出来了，9900 公斤。

总计重量：车架 9900 公斤，引擎 230 公斤，车窗 360 公斤，轮胎 80 公斤，油箱 230 公斤，车座位 690 公斤。加起来，合计 11 490 公斤，也就是约 11.5 吨。对于一辆校巴来说，这个数字看起来像是合理的。

实际答案：根据大小的不同，一辆校巴的重量可以在 4.5 吨到 14 吨之间。而一辆载客量为 62 人的标准校巴（我们假设的这一辆大致如此），不计油箱大约重 9 吨。

样题 8：美国每年卖出多少个篮球？

一般而言，会买篮球的有学校、（非学校的）球队和家庭等等。我们先计算每个团体拥有多少个篮球，再估算出他们的更换率。

假设成年人购买篮球供自己用（而非给他们的孩子玩）的数量可以忽略不计。

家庭：美国有 4500 万 6 岁到 18 岁的孩子。假设未育有孩子的家庭拥有的篮球数量少到可忽略不计，并假设平均每个家庭育有 2 个孩子。这就意味着有 2000 万家庭育有孩子。

我猜想 1/4 育有孩子的家庭拥有一个篮球，而且平均一年更换一次。那就是说，这些家庭一年会购买 500 万个篮球。

学校：我的高中有 500 名学生，拥有 50 个篮球。由于经常使用，这些篮球一般都会在 3 个月内被更换。一学年有 9 个月，也就是说规模为 500 人的学校一年会购买 150 个篮球。

不过话说回来，我就读过的高中在经费上比大多数的美国高中都要高。规模为 500 人的学校一年会购买 50 个篮球，这个数字听起来相对真实一些。

说实话，这个数字还需要再往下调整，因为小学生用不了这么多篮球。那么我们就假设每 500 名学生拥有 25 个篮球吧，也就是说平均每 20 名学生拥有 1 个篮球。

4500 万学龄学生，意味着学校每年会购买 200 万个篮球。

球队：美国有各种各样的球队：儿童篮球队、职业篮球队、成人校内运动队、学校校队，等

等。上一部分，我们已经将高中计入，所以这一部分不需要重复计算。

- 儿童篮球队：假设 1%的孩子加入了篮球队，一支篮球队有 10 个孩子，那么美国总共就有 45 000 支儿童篮球队。如果平均一支球队每年购买 10 个篮球（平均 1 名球员 1 个），那么这就有 45 万个篮球。
- 职业篮球队：可忽略不计。
- 成人校内运动队：可忽略不计。
- 大学球队：美国有 1500 万处于上大学年龄的人口，其中有 5%是高校运动员，这其中又有 5%是篮球运动员。所以美国总共就有 37 500 名高校篮球队员。如果我们再次假设平均 1 名球员就有 1 个篮球，那么这就有 37 500 个篮球。

也就是说，球队每年会购买 50 万个篮球。

我们已经计算出了家庭购买 500 万个篮球，学校购买 200 万个篮球，球队购买 50 万个篮球。所以，总的说来，美国一年总共卖出 750 万个篮球。

样题 9：美国人每年花费多少钱在理发上？

首先要注意到，理发的价格可以是天差地别的。在一些高级美发厅，一次理发加上染发（女性经常这样做）可能会花费 200 美元甚至更高价格。另一方面，男性在便宜的理发店理一次发可能才 10 美元。我们需要在解题的过程中考虑到这个问题。

假设男性平均一年理发 12 次（平均 1 个月 1 次），女性平均一年理发 5 次。

我想 2/3 的人（包括成人与儿童）都由专业理发师理发，剩下的人在家理发或由朋友理发。也就是说，1 亿男性和 1 亿女性由专业理发师理发。

因此，平均每年会有 12 亿次男性理发和 5 亿次女性理发。

就我本人的观察而言，我几乎没见过男性去高级美发厅理发，所以可以假设这 12 亿次男性理发都是在花费较低的理发店。如果这种理发店平均一次的价格是 20 美元，那么男性理发每年就会花费大约 240 亿美元。

更多的女性会去高级美发厅做头发护理。假设 20%的女性选择均价为 100 美元的高级美发厅。这意味着女性的平均理发价格是 36 美元。

因此，女性理发每年花费约为 180 亿美元。

合起来计算，美国人每年理发的花费约为 400 亿美元。不过，我觉得这个数字显得有点高。我猜想，还有 25%的人并不理发，有因为谢顶的，也有因为是幼儿的，诸如此类。所以，最终答

案可能是接近 250 亿[①]。

样题 10：Facebook每年在广告上赚了多少钱？

我记得不久前读过一份报告，称 Facebook 目前已有大约 10 亿用户。然而，当中有许多可能并不是活跃用户。假设这些用户中有 50%是真实的活跃用户，且活跃用户平均 1 天登录 1 次 Facebook 吧。

这意味着 Facebook 每天会有 5 亿人次的访问量。假设用户平均访问 1 次大约持续 10 分钟、10 个 PV（页面访问量），那么平均 1 天就有 50 亿个 PV。

如果一个页面有 4 则广告，那么 Facebook 平均一天就给用户展示了 200 亿则广告。

印象中谷歌上搜索式广告的点击率应该是 2%，但我还知道，搜索式广告的点击量要高于页面嵌入式广告的点击量。假设它们之间有 10 倍的差距，那么就得到了嵌入式广告的点击率是 0.2%。

也就是说，平均 1 天的广告点击量是 4000 万。

平均每点击一次广告，能转化成多少收入？

据我所知，谷歌搜索式广告的点击收入一般都不低于 5 美分，甚至有达到 10 美元的。显然，这是一个巨大的范围。根据我和广告打交道的测试，我发现我每次点击要付出大约 25 美分。这个数字恐怕还低于市场平均值哩！因此，还是假设每次点击能收入 50 美分吧。

现在假设 Facebook 的广告和谷歌广告一样，但是事实上它们很可能是不对等的。

但是目前就先这么着吧。

1 天 4000 万点击量，1 次点击收入 50 美分，可以得出 1 天收入是 2000 万美元。也就是说，一年的收入大约是 70 亿美元。

这听起来应该还算靠谱，再看看人力薪酬成本。如果 Facebook 有 5000 名员工，平均一名员工年仅 10 万美元，那么就有 5 亿的薪酬成本。Facebook 的收入是其薪酬成本的 10 ~ 20 倍，这听起来尚算合理。

实际答案：2012 年，Facebook 的年收入是 51 亿美元，其中大约 43 亿来自广告收入。

---

① 原文是 25 百万（25 million），即 2 千 5 百万，但根据上文判断为原文错误，故译本处理为 250 亿（25 billion）。
——译者注

第 14 章 产品问题

杰西卡到苹果公司面试一个 iTunes 的职位时，面试官问了她一个简单温和的问题："你认为 iTunes 怎么样？如果要你来设计，你会做什么改变？"因为她平时非常热爱音乐，所以对这个问题作了充足的准备。

她立马滔滔不绝地列出一堆她希望落实或修改的功能：更好用的键盘导航、更简便地取消下载、iTunes Store 搜索体验加强等。她越说越多。

她提了很多意见，而且总的来说，这些想法都相当不错。然而，最后她并没能进入下一轮面试，她认为是因为面试官觉得她太苛刻。

其实不尽然。

喜欢或讨厌一个产品没有什么不好，但更重要的是你得正确处理这些问题。

14.1　关于产品问题

产品问题是产品经理面试的核心和灵魂。这类问题直接和产品经理的工作相关：设计产品、制作产品和改进产品。产品问题通常都以三种常见的形式出现：

(1) 如果是你，你会如何设计某产品？比如说，你可能会被要求为失明者设计一个闹钟。
(2) 关于某种产品，你会如何进行改进？
(3) 你最喜欢的产品是哪一个？为什么？

这些问题听起来虽然不尽相同，但它们都有一个重要的相似点：你需要理解这些问题的最终目的。这个目的可能是要为用户设计出最好的产品，也可能是为了增加收入或其他意图。

14.2 类型 1：设计一个产品

这类问题远没有听起来那么开放。遇到这类问题，你得表现得像一名出色的产品经理：使用结构化的方法，优先考虑用户需求。

记住：不是你想要产品怎么样，而是用户想要产品怎么样。

14.2.1 解题方法

针对此类问题，我们提供了一套很得力的解决方法，但这并不是唯一的思路。好的方法有这些共同点：问适当的问题，理解和评估问题的目的（常常是优化用户体验）以及应用结构化方法来达到目的。

步骤 1：通过提问来理解问题

开始回答问题之前，首先确保自己确实理解了这个问题。某些情况下，它未必就是你认为的那样。

比如说，你被要求“设计一支笔”。这是一个简单直接的问题，对吗？其实也不尽然。

笔可以是：

- 油性马克笔，用来为衣物着色，不会因为洗涤而褪色
- 使用神奇墨水的钢笔，只有在特殊的光照下才会显示出字
- 专为宇航员设计、在太空使用的笔
- 儿童在浴缸里使用的笔
- 潜水员专用的笔

显然，上面说的这些人需要的是不同的笔。他们需要不同的尺寸、颜色和功能。

面试官出这种题，莫非是想要你？是的，从某种意义上来说是这样的。但是，产品经理若是未理解产品的设计目标就贸然开始做这个产品，最终可能会与用户的期望背道而驰。

步骤 2：使用结构化思维

面试官想要看到的是一个结构分明的思维。而要表现出这一点，最简单的方法就是给出一个结构分明的答案，并时刻声明你的思路到了这套结构的哪一个节点上。举个例子，你可以这样说：“首先，我想要说说设计这个产品的目的。接着，我会列出一些可行的功能点。最后，我会结合产品设计目的对这些功能点进行评估。那么，现在我们开始讲这个产品的目的……”

这样，你就能向面试官传达一种信息——逻辑清晰的解答能力。这种方法对于保持自己和面试官的思路清晰也有帮助。

步骤 3：明确目标用户和消费者

既然已经理解了问题本身，那么接下来就应该搞清楚这个产品的目标用户和消费者都是什么样的人。如有需要，还应该向面试官提出更多问题。

有些情况下，目标用户和目标消费者并不是同一人群。消费者是愿为产品付费的人，用户是使用产品的人。有时候，用户并不是单一人群，也可能是不同的用户群。

样题：设计一个孩子专用的计算器

在这个案例中，面试官已经将用户指明了。真的指明了吗？当然，世界上有很多孩子，但是他们并不都是一样的。孩子并不是唯一的用户。

事实上，我们有以下潜在用户或消费者。

- **孩子**　孩子是这个计算器的主要用户。你需要知道这些用户的年龄层。一个为 7 岁儿童设计的计算器和一个为高中生设计的计算器可大不相同。
- **教师**　如果这个计算器是在学校里作学术用途的，那么孩子的老师大概也需要使用这个计算器，或至少懂得它的使用原理。
- **父母**　父母在辅导孩子功课时，可能也要用到这个计算器，他们还可能就是为计算器付费的人。这就使得父母既是用户又是消费者。

根据计算器的设计类型，可能还会有更多潜在用户和消费者。比如说，如果专门为某本教科书设计这个计算器，那么教科书的出版商也包含在目标用户之内；或者，如果这个计算器需要在教室内使用，那么学校或教育部门可能就是我们的消费者。这些都需要仔细考虑。

样题：设计一个更好用的炉子

正如前面说到的“设计一支笔”的例子，问题所指的“炉子”可能和我们想象中不一样。是家庭中使用的，还是大型餐厅使用的，或是小孩子玩的？我们需要提问，来明确问题所指。

首先，我们不妨假设，这个产品没有什么花哨的东西。我们只是要设计一个为家庭所用的普通炉子即可。不过尽管如此，我们还是需要知道目标用户的不同类型。可能会包括以下人群。

- **新手厨师**　这些人新接触烹饪，可能想要简单的功能。
- **高级厨师**　这些人对烹饪很在行，可能想要更多高级功能。
- **孩子**　就算孩子并不使用我们的炉子，他们也是家庭中的个体，会常常在我们的产品周围晃悠。一个好的家用炉子不能给孩子带来安全问题。
- **长者或残障人士**　由于长者或残障人士的身体活动有所不便，他们可能会有一些稍微不同的需求。他们可能会有特殊的烹饪要求。

这些人群的需求都不一样。

为用户考虑

遇到每个问题，都要想象产品的使用场景，以及除了用户之外，还有什么人会与产品互动。以下是一个很好的方法。

- 儿童产品的用户可能会是孩子、他们的父母和老师。父母或学校可能是消费者。
- 保健产品（包括为残障人士设计的产品）的用户可能是病人、医生和保险公司。
- 运动产品的用户可能是运动员和他们的教练。
- 专业类产品（如会计软件）的用户可能是专业人士、他们的助手和同事。

我们设计产品，得考虑所有目标用户，因此必须要询问面试官以明确目标用户的范围。

步骤 4：有哪些使用案例？用户为什么要使用这个产品？他们的使用目的是什么？

为每种用户（如果有多种用户的话）列一个使用案例的清单。这个清单要列出用户使用这个产品各种可能的使用目的和使用场景。

比如说，如果我们要为年长者设计一个钥匙圈，使用用例可能包括：

- 在钥匙圈上找到正确的钥匙，打开家门、车门，等等；
- 在钥匙圈上再扣上一把新钥匙；
- 从钥匙圈上移除一把钥匙；
- 在提包里找到钥匙圈；
- 在住所里找到这个钥匙圈。

你需要评估这些使用案例，独立评估或和面试官讨论都行，并决定要依据于哪些案例来设计产品。可以说所有情况都是重要案例，也可以决定哪一些比较重要，哪一些不那么重要。

你也可以考虑，哪些使用目的更高级。你不仅可以考虑用户使用这个产品来做什么，还可以考虑他们为什么要使用这个产品。用户的潜在动机是什么？比如说，用户对钥匙圈的潜在动机可能是独立生活。

其次，你得说服面试官，你的产品能通过激发用户的潜在动机，满足他们的使用目的和使用场景来改变世界！

步骤 5：现有产品能否良好地满足这些使用需求？这些产品是否存在明显的缺点？

逐个复核使用案例，评估现有产品或解决方案如何满足这些案例。用户会觉得这个产品最大的问题是什么？而这些问题就是你将要在设计中注意的。

如果有不同用户群（比如老人与他们的护工），我们可能需要分别评估他们的使用用例。

在很多情况下，尤其是当你被要求“为某个人群设计某种产品”的时候，仔细想想这群人有

何特殊之处，这个思考可能会有帮助。举个例子，一位老人在身体行动和灵敏度上可能有所不便，但是长者不仅仅只有生活不便这个特点。对产品来说，这个人群也有其他特质。他们可能会非常在乎与家人的关系，或者会把医疗保健或安定的生活放在首位。我们在设计中需要时刻记住这些因素。

步骤 6：什么功能或修改能改进这些缺陷？

一直到现在，我们都在评估现有的问题和需求。花时间确定问题所在是很好的。这能帮助我们想出一个为他们的需求量身定制的解决方案，而不是一意孤行只满足自己的需求。

很多情况下，我们得同时考虑多个用户需求来解决问题。比如说，在钥匙圈上多扣一把钥匙和在钥匙圈上移除一把钥匙的解决方案是十分相近的。

有个好办法可以处理这一部分，就是向面试官提出几个想法，并询问他想要对哪一个想法进行深究。

一定要明确地将你要设计的功能和用户的使用需求或目的紧密挂钩。要让面试官清清楚楚地知道，你想出来的主意都是消费者关注的，而不是你自己凭空想要的东西。

如果你开始技穷了，那就回到使用需求清单去看看，让自己再来点更有创意的东西。要是面试官确实很希望你还有更多想法，就问问她吧，看看是不是还有什么是你没有充分探究的。

这也是一个使用白板作记录的好时机。

步骤 7：总结

在面试的最后一个环节，如果能将你的解决方案给面试官作一个回顾总结，应该会有帮助。要是你恰好侃侃而谈了好一会儿，这个总结就尤其重要了。你可能已经为问题提出了很多个解决方案了，而面试官此时可能正一头雾水。

如果这个时候你还没用过白板，那么是时候用上了。

14.2.2 样题：为失明者设计一个闹钟

我们逐步骤分析这个问题吧。注意，这只是一个可能的解决方案，要解决这个问难题其实可以有很多方法。

步骤 1：提问，以理解问题

在这个问题里，我们被清楚地告知了目标用户。然而，这个信息只有一半是正确的。

用户是哪种类型的失明者？首先，很多失明者可以感受到光线，甚至可以看到模糊的影。其

次，失明者还可以分成儿童、成人和长者。他们可能还有其他的残障问题。如果我们要为特定的失明群体设计这个产品，那么这些问题就会对产品造成影响了。

同时，我们还应该搞清楚，失明者在什么地方使用这个闹钟。这是一个家用闹钟，还是旅行用的闹钟？它是一个实体闹钟，还是说面试官可能想要你设计一个为失明者所用的手机应用？

我们不妨假设面试官确认过了，这是为完全失明者设计的，而且是为成年的失明者设计的、专门在家使用的实体闹钟。

步骤 2：使用结构化思维

解决这个问题的方法，本质上就是我们展示给面试官的结构化思维。

> 那么，既然已经清楚这个问题了，就可以将其拆解为几个部分。首先，我会考虑用户是哪些人，他们为什么要使用这个闹钟。接着，我要将现有的闹钟产品和这些使用需求作对比，看看它们之间的差距是什么。最后，我会对如何填补这些差距进行讨论。

在对问题进行逐步破解的时候，要让面试官清晰了解你的步骤。你可以说一些诸如此类的话："现在我们已经明确了用户群，下一步我们来评估以下现有的闹钟产品。"

> **注意措辞**
>
> 要注意你提问时的措辞。诸如"失明者平时做什么工作"这样的问题听起来就好像你在要求面试官帮你解决问题似的。然后，如果你改变一下措辞："不知道一般失明者平时所做的事，有没有我们熟悉的呢？"这样听起来就好得多，就像你在尝试和面试官协作完成任务。

步骤 3：明确目标用户和消费者

目前，我们已经有了一个明确用户群：完全失明者。

还有什么人可能会使用这个闹钟？一个失明者可能身边有许多非失明者，比如他们的配偶、儿女或护工。你能想象你的配偶买了一个你无法关掉的闹钟吗？没错！一个为失明者设计的闹钟还是需要为非失明者考虑，让他们也能合用。

步骤 4：有哪些使用需求？用户为什么要使用这个产品？他们对产品的期望是什么？

这里的核心用户的用户场景就是要睡醒，早晨醒来、小睡醒来。这就意味着他需要检查闹钟是否设定好了，配置时间、启动闹钟、被闹钟唤醒、设置闹钟重响和关闭闹钟。

失明者要使用这个产品，还有没有其他原因？当然有！

- 他可以用这个闹钟来了解时间。
- 他可以用这个产品来计时（如烹煮鸡蛋时）。毕竟不是所有的电器都能适应他的残障，所以他可能会经常使用这个闹钟来补偿这种失落。
- 他可以用这个闹钟来提醒自己吃药。在这个场景里，他应该还需要一些重复闹钟的功能。
- 他可能用这个闹钟来听音乐或听收音机，很多人都会用闹钟这样做。

假设我们要设计的基础功能是为了满足睡眠唤醒的需求。

总的来说，查看当前时间当然也是使用闹钟的需求之一，但对失明者来说，这可能不是一个必要的功能。人们使用闹钟查看时间，是因为这样做非常方便（因为闹钟就放在那里，隔着房间看一眼就知道了）。除非我们能做出同样方便的体验，否则失明者不会为了知道时间而使用这个闹钟。他们肯定还是更倾向于以往使用的方法来获知时间。

步骤 5：现有产品能否良好地满足这些使用需求？这些产品是否存在明显的缺点？

标准闹钟的任何属性，都会直观地展现出来。我们知道闹钟上一般有一个指示灯，指示我们闹钟的通电状态。我们只需要看一眼闹钟，就能知道闹钟有没有通电。基本上，闹钟唯一不需要用眼睛看的用途，就是唤醒功能。

这对失明者带来了不少挑战，因为他没有办法看到闹钟。

- **开启/关闭闹钟**　闹钟有一个按钮可以切换闹钟的开启状态；按“设置闹钟”按钮来触发状态。如果你能看到闹钟的设置情况，那么就没什么障碍，但是对失明者来说，这可能是一个问题。切换某物的状态，只有在当前状态非常明显的情况下才能有效地起作用。
- **确认闹钟是否通电**　电子产品在很多情况下会被断电。我们只需要给失明者设计一个能检查闹钟是否通电的方法就够了，他们大概不会忘记在睡前要来检查这个设置。我们干脆这样，就让断了电的闹钟有一个明显的提示吧。
- **检查是否设置了闹钟**　标准闹钟有一个指示灯可以表示是否启用了闹钟。由于失明者看不到指示灯，这个指示灯对他们来说是没有意义的。
- **设置闹钟**　标准的闹钟设置是通过“+”“－”按钮来指定时间，直到调整正确的时间。失明者看不到闹钟，所以即时可视的时间调整对他们来说也是没有意义的。
- **设置/查看当前时间**　虽然失明者不需要看时间的功能，但我们还是需要一个正确的当前时间。由于标准闹钟直接展示时间给使用者，这一点对失明者来说也不能起到很好的作用。

现在，我们需要解决这些问题。

步骤 6：什么功能或修改能改进这些缺陷？

我们希望这个闹钟很容易操作。复杂的设计都是不好的，无论用户是不是失明者。

设计方案 1：播放提示音

我们的设计中最需要围绕的问题就是他们无法看到可视化显示界面。我们可以使用一个非常简单的方法，就是在可视界面之余，增加播放音频提示。这样设计有一个好处：建立提示音播放机制是很简单的事情，而且可视界面部分能方便非失明者使用。

然而，这样做也有一个重大缺陷：它可能会吵醒失明用户的伴侣或配偶。不过，我们目前还是考虑先使用这个方法。

开启/关闭闹钟 + 确认闹钟是否通电

和标准闹钟一样，这个闹钟也有一个按钮可以开启和关闭闹钟。当闹钟指示“已开启闹钟”或“已关闭闹钟”时，我们会播放提示音。这样做还能方便用户得知闹钟是否已意外断电。

检查是否设置了闹钟

为了保持产品简单易用，我们可以重复使用上面说到的开启/关闭按钮，而无需为了这个功能增加一个按钮。

设置闹钟重响

为了能设置闹钟重响，会有一个单独的“贪睡”按钮，就和普通闹钟一样。

设置闹钟时间

要设置闹钟时间[①]，我们可以提供“+”“－”按钮来修改时、分。每次按下按钮时，它就声明一次当前时间。这样做的话，每次按下按钮之后的反应时间可能会稍久一些，尤其是修改分钟数的时候。那么，我们也可以这样设计：长按“+”、“－”按钮时不播放提示音，松手时才播放。

设置当前时间

设置当前时间基本上和设置闹钟时间的方法一致。

设计方案 2：限制性播放提示音

如上述所说，使用音频提示最大的问题就是它可能对同居一室的人造成打扰。对此，我们可能需要设计一个更高级的闹钟，让它最常用的功能不依赖于音频提示（尤其是失明者伴侣睡觉时使用的功能）：开启/关闭闹钟、设置闹钟时间。

开启/关闭闹钟

这里就不播放提示音了。我们可以使用物理开关（就和照明开关一样）来开启或关闭闹钟。

① 原文是“当前时间”（current time），但根据上下文，判断为原文错误，故译本处理为“闹钟时间”。——译者注

但这样做也有风险：闹钟可能意外断了电，而用户却不会知道。

不过，还是可以使用一个物理按钮来切换闹钟状态。提示的不是音频，而是以一次震动表示“已开启闹钟”，两次震动表示“闹钟已关闭”。

检查是否设置了闹钟

同理，我们直接使用开启/关闭闹钟的按钮即可。闹钟会以震动的方式告诉用户，闹钟是开启的还是关闭的。

设置闹钟时间

为了作设置时间，闹钟的侧边有若干排按钮。每个按钮会有盲文和着色文字表示数字，以便失明者和非失明者都能使用。由于此处使用按钮来设置时间，我们就不需要播放提示音了。

时	12	1	2	3	4	5	6	7	8	9	10	11
分	0	1	2	3	4	5	6	7	8	9	上午/下午	
	0	1	2	3	4	5	6	7	8	9		

按下“上午/下午”按钮表示设置为上午，松开则设置为下午。

这样设计的话，可能比事实所需多出太多按钮了。是不是真的有人非要设置他们的闹钟为像“上午 9:23”这么精确的时间呢？大概没有吧！实际上，设置为 9:25 或 9:20 就好了。

好，那么我们就移除掉一些按钮，让产品简单一些吧。简约的设计就是美。

时		
12	1	2
3	4	5
6	7	8
9	10	11
上午/下午		

分		
:00	:05	:10
:15	:20	:25
:30	:35	:40
:45	:50	:55

或者，也可以使用循环转盘来代替设置分钟数的按钮。然而，用来代替时数按钮就不够理想了，因为用户可能会一不小心就设偏了整整一个小时。

步骤 7：总结

根据我们对“吵醒用户伴侣”这个问题的关注优先级，可以决定需要或者不需要播放提示音。此时，可以向面试官说明一个不打扰用户同床伴侣的闹钟的市场销售潜力。

第一个设计方案使用音频提示，基本上是复制了可视闹钟的功能。为了让非失明者能使用这个闹钟，我们还是会有界面显示。但是这个方案会因闹钟或其他任何设置而发出大声响。唯一不会引起明显声响的设置就是长按“+”“–”按钮修改时间的时候。

如果我们关注吵醒用户伴侣的问题，可以去掉音频提示，用震动（指示闹钟状态）和物理（设置时间）按钮来代替。

不管是哪一个方案，都要保留可视界面，以保证非失明人士也可以使用这个产品。

14.3 类型 2：改进一个产品

这类问题的措辞可能会很笼统（“挑一个产品。如果要你来设计，你会怎么改进这个产品”），也可能会指定一个产品（如“如果要你来设计，你会怎么改进某产品”）。无论是哪一种说法，解决这类问题的窍门就在于找到并理解这个产品最大的症结所在。

可能你会注意到，这类问题的解决方法和“设计一个产品”类问题非常相似。没错，是这样的。关键差别就在于你要分析一个现有的产品，评估它的问题，并对它作出改进。

同样，结构化思维在这里非常重要。你不必非得强调你的结构，但至少应该以一种有条理的方式来解决问题。使用连接词是一个很好的范例：“首先我要理解这个产品的设计目的，接着我会分析它的缺陷，并关注这些缺陷的解决方案。那么，这个产品的设计目的是这样的……”

步骤 1：这个产品的设计目的是什么？

首先，你需要知道这个产品的终极设计目的。它能为用户解决什么问题？

举个例子，你能在 Facebook 上发布状态更新、分享照片。但这并不是这个产品设计的初衷，它的原始目的是连接你与家人朋友，向他们分享你的生活。

注意，一个产品可能会有主要目的和次要目的。Facebook 的主要目的就是令消费者与家人朋友连通，但商业用户还有另一个目的，那就是和老客户、潜在客户进行接洽。

步骤 2：这个产品面临着什么问题？

接着，你需要评估这个产品面临的问题所在。

- 是否需要扩大用户群？如果需要，是通过打开新的市场，还是通过扩大原有市场来扩大用户群呢？
- 是否需要提高收入额？如果需要，是要提高用户平均收入还是增加付费用户的基数？
- 是否需要提高用户粘性？
- 是否需要提高游客的转化率（转化为注册用户）？

要评估产品问题，先想一想这个产品的设计流程。这个产品优先关注哪些地方？

当然，你也可以就这个产品的关键问题向面试官询问，但你应该不想让他觉得你自己没有能力弄清楚产品的问题吧！因此，你可能需要凭经验对需要改进的地方作出猜测，然后和面试官进行确认。

步骤 3：你会如何解决这个问题？

第三步，想出若干可能会用来解决这个问题的方法，并讨论这些方法的优劣。同样，要对这些方法中需要作出的取舍保持开放的心态。

你可能会想出一些大胆、颠覆性的改变，也可能只是一些微小的迭代改进。

这两种改进方法（以及介于两者之间的任何方法）在真实世界中都有巨大的价值。

然而，一些公司可能会基于公司规模、风险耐受度、预算或企业文化对这两种方法作出取舍。如果你知道这家公司更倾向于哪一种方法，就可以专注讨论这一种方法。否则，就应该对两者都作出相应的建议：

> 我们可以作出若干快速的修复，来缓解这个问题。但是，如果我们愿意冒更大的险，也可以考虑追加一些功能点。
>
> 至于快速修复，可以这样解决……

这样做可以证明一点：你很清楚产品的大型修改能带来大收益，同时也会带来高风险。一名出色的产品经理需要平衡这两个方面。

步骤 4：如何实现你的解决方案？

如果你建议的方案不够直接明显，那么就要和面试官讨论一下如何实现你的解决方案了。最大的技术挑战和商业挑战是什么？如何降低解决方案相关的成本或风险值？举个例子，可能需要用一个小用户群对你的解决方案进行测试，或者展示你的产品原型方案。

很多创业公司会看重一个好胜的、能充分利用有限资源的产品经理。让面试官看到你这方面的能耐吧！

步骤 5：如何确认你的解决方案是行之有效的？

一名出色的产品经理很清楚，他并不永远是对的，而他的方案只不过是一些有根据的臆测而已。因此，确认方案是行之有效的，这是产品经理日常的重中之重。

和面试官讨论一下，如果你的方案起作用了，你将需要知道哪些衡量标准。举个例子，如果

你提议发送如“买下产品 X 的顾客也买了产品 Y”的电子邮件以提高用户平均收入，可能你就要追踪这些衡量标准：电子邮件点击量，商品 Y 的外部直接跳转地址点击量，来源于电子邮件的商品 Y 未来购买量以及其他产品的购买量。

同时，可能你还要想一些方法，来确保这些电子邮件不会对用户造成骚扰，导致他们取消所有的邮件订阅。

14.4　类型 3：你最喜欢的产品

一般而言，至少一位面试官会问你最喜欢什么产品以及为什么喜欢这个产品（请看本节的准备工作贴士）。

这些问题也类似于“改进产品”，只不过方法是相反的：不是让你谈论这个产品有什么问题，而是讨论你为什么喜欢这个产品。

一般遇到这种问题，人们都会谈论自己使用的产品。不过，当然也可以选一个自己不使用但热爱的产品。举个例子，父母可能会认为一个儿童玩具设计得很棒，是他们“最喜欢的产品”，哪怕他们本人都不曾使用。

有些产品可能并不适合作为这个问题的答案。要是想不出能引起面试官兴趣的特点，那你可要当心了，相比之下，不如先不要谈论这个产品。

同样，你也不应该仅因为产品有趣就挑选它来讨论，而是应该选一个和你有亲密关联的产品。你要将你对这个产品的热情表现出来。不要因为某个产品听起来能得高分就挑选它，而要挑选一个适合自己的产品。

同理，你对这类问题也要表现出结构化的思路。这个问题并不要求你强调你的结构（但是你也可以这样做，只要听起来自然就行），但你仍应以一种有条理的方式来回应问题。任何情况下，优秀的沟通技能都是很重要的。

步骤 1：这个产品为用户解决了什么问题？

你或者你熟悉的人应该就是这个产品的目标用户。你不仅需要思考这个产品能做什么，还要思考用户使用这个产品的目的何在。

以分析个人财政情况的产品为例吧，比如 mint.com。分析个人财政情况是这个产品的功能，但这不是这个产品的目的。它的目的应该是通过让你清楚自己的开销，来帮你达到省钱的目的。难道你上个月真的想在食物上花 1000 美元吗？

你可以谈论多个使用目的，但也别太过火。如果列举了太多使用目的，你的讨论内容可能就

失去了重点，更何况你很可能只是在讨论功能，而不是目的。

步骤 2：这个产品如何满足这些使用目的？是什么使得这个产品很“干净”？这个产品的什么地方使得用户爱上了它？

现在你应该谈论这个产品如何满足这些使用目的。肯定有什么特性，让这个产品在某方面特别强大。

举个例子，它可能有丰富的用户数据足以支撑它进行一些强大的分析。或者它可能有一个出彩的用户界面，每个按钮都被安排到了恰到好处的位置上。

除了在满足用户使用目的方面非常优秀之外，这个产品可能还有其他吸引你的地方。例如一个精明的、与用户双赢的收入模式，或是一个依赖众包模式削减成本的方法。谈论这些细节，也是可以的。

对很多产品而言，也有可能是一些情感上的联系使得用户爱上它。有些产品真的很受消费者喜爱，它们就是人们大肆追捧的东西。为什么会这样?

步骤 3：它和同类产品相比起来如何？

每个产品都有替代的产品，即使是那些看起来像是市场新秀的产品。往狭义和广义想，有哪些产品是这个产品的“竞争对手”。直接对手可能是能实现同一目的网站，间接的对手则可能是能具有同一功能的实物产品。

换句话说，总有一个原因使得你没有使用这些竞争产品。那么会是什么原因呢?

步骤 4：如果要你来设计，你会如何改进这个产品？

你不必非得立刻谈论这个问题，但这是面试官很自然就会发问的问题。你可以使用上一类型的解决方法来拿下这个问题。

样题：你最喜欢的网站是哪一个？

如果是我，我可能会这样回答：

“我是 Quora 的超级粉丝。Quora 是一个问答网站，用户可以发布问题寻求帮助，通常会有一些相关主题的专家来回答问题。比如说，有人可能会问：‘更换轮胎最好的办法是什么？’或‘创业公司应该等候时机再筹集风险投资吗？’，甚至是‘住在牧场里是什么感觉？’网站上的问题五花八门，从‘帮我解决问题’到‘你对某事有什么看法’，

再到‘请分享这方面的经验’，一切都应有尽有。

“我热爱 Quora 的原因之一就是它在解决专业问题时的高效。基本上任何行业的专家都有，提供专业意见，能够给出极其详尽、深思熟虑的答案。

“它在很多问题上给了我帮助。比如说，我在处理一个产品的翻译版本时，我需要知道日文版本和英文版本页数上的差别。我得到了几个很棒的回答，它们提供了好几种不同的方法。不是开玩笑，这些人真的很专业。

“还有一次，我想知道在我家附近有哪个咖啡厅的环境适合学习。最后发现很多本地人和我一样，就是从 Quora 上发现了那家我最喜欢的咖啡厅。

“然而，在解决专业问题之余，我发现我还很喜欢使用这个网站。事实上它是一个基于学习和探索的社区。你可以认识网站上的人，评价他们的回答，因为你可以信任他们。我从来就不是那种在 Quora 上‘交朋友’的人，但是在 Quora 上我确实感觉我熟识这些人。它是一个真实的社区。

“我想它是少数能将学习和乐趣连结起来的网站之一。它是一个问答型的网站，每当我偶然发现一个自己从来不感兴趣但却很想知道的话题，就会到上面去询问别人。

“在我个人使用的经历中，很多时候它都代替了谷歌，能够帮我搜到我想要的答案。想想看，其实通过搜索引擎来解决问题，是有几种缺陷的。

“第一，你会使用关键词来搜索，即便要解答一个问题，也同样如此。你是希望找到一个页面，能为你提供全面纵览的视角来解答你的问题。这并不是一种以解决方案为导向的搜索方式，通常效率很低。

“第二是信任问题。你可能不信任搜索结果中网页作者的证据。就算你信任他，他们也可能在自己的专业领域里把问题搞错了。

“Quora 允许用户对问题答案以评论的方式进行反馈（也有“点赞”和“反对”的形式），这是一个判断答案准确性的好办法。而谷歌搜索答案的信誉度多是以 PR 值（网页评级）的形式表现的，那可是相当不严谨的一种做法。

“第三，有很多信息还是很难通过普通网页和博客来获取。网站会更偏向于某些有专业能力又愿意创建页面的作者。我们能很容易找到技术方面的支持，但如果我想了解如何演绎一部话剧中的某一角色呢？可没那么容易。

“当然，我仍会在网上搜索答案，但我发现我越来越经常使用 Quora 来解决问题，并且它也是我浏览和学习新知识的地方。”

虽然我在回答中批评了 Quora，但我也做了一些别的事情。

- ❑ 我暗示了自己是个热爱学习的人。
- ❑ 我谈论了 Quora 吸引我的原因：它以问题为基础，而且是一个社区。这就使得这个产品有能力引起一些用户对它产生情感寄托。
- ❑ 我谈论了一些不那么明显的网上搜索缺陷。

当然，这个案例仅供参考。挑一个最吸引你的网站或产品吧。

14.5　准备工作

熟能生巧。你应该多加练习这类问题，自己练习也好，和朋友一起练习也好。另外，在面试之前就对这类问题打好了腹稿，是很重要的。

步骤 1：选择产品

参加面试的时候，你应当准备好深入谈论以下这些内容。

- 一个线上产品。
- 一个“线下”实体产品。面试官很喜欢要求候选人评估一个实体产品，以此来刁难候选人。
- 你最近购买的一款产品。
- 你最喜欢的一个产品或网站。
- 一款你认为设计得特别好的产品。
- 他们公司或团队的产品。

很多公司会问候选人：“你购买的最后一个产品是什么？”其实不用非得真是最后一个产品不可，只要这个产品是新近推出的就可以。

上面枚举的这些产品可能会出现重复。比如，你选择的线上产品可能就是你最喜欢的产品。但是作好准备，谈论更多的产品可能是颇有价值的。

步骤 2：理解产品的关键衡量标准

针对每个产品，思考并搞清楚它的关键衡量标准是哪些。这些衡量标准可能会包括以下内容。

- **用户/下载流量**　这个产品有多少用户？这个产品是如何吸纳用户的？
- **用户转化率**　这个产品在将游客转化为注册用户、将免费用户转化为付费用户、将付费用户转化为高付费用户的转化率有多高？
- **用户推荐率**　用户会将产品介绍给他人吗？介绍频率怎么样？这个产品能被急速推广吗？
- **用户粘性**　用户是否积极使用这个产品（发布、评论、玩耍，等等）？使用频率怎么样？
- **回访率**　用户是否经常回访？有多少用户会隔一段时间再来访问？对某些产品来说，一个月一次访问已经很好了。但也有一些产品需要用户更频繁地访问。
- **收入**　这个产品如何赚钱？它赚多少钱？
- **成本**　这个产品的成本体现在哪些方面？是否需要物理材料？它的开发成本是什么？是否需要大量的售后支持和销售成本？

将这些衡量标准从头到尾细想一遍，看看哪些是对这个产品有决定性的，哪些是这个产品最强大的，还有哪些是这个产品还在苦苦挣扎的，这很重要。如果是你，你会对这些衡量标准作出什么改变？

另请参见本书第 15 章的“产品衡量标准”。

步骤 3：分析各个产品

目前，你应该已经清楚了各个产品的衡量标准（想办法搞清楚），现在可以开始通过以下各方面分析产品了。

- ❑ **用户群和使用目的** 产品的主要用户群是什么人？ 他们的使用目的是什么？这个产品如何帮助用户实现他们的使用目的？
- ❑ **优势** 这个产品的哪一个衡量标准最强大？它是否拥有很多用户？它是否有很强的用户粘性？
- ❑ **挑战/重心** 这个产品面临的主要挑战是什么？它是否还在挣扎着吸引用户来注册？它是否正在积极将免费用户转化为付费用户？
- ❑ **为什么？为什么？为什么？** 为什么这个产品在某个方面特别强大（或特别弱）？
- ❑ **优先偏好与价值观** 这个产品或这家公司优先关注产品的哪些方面？比如说，苹果公司沉湎于用户体验。而一个企业级产品的公司可能会优先考虑安全性能及稳定性，而不是华丽外表。
- ❑ **竞争对手** 这个产品有哪些竞争对手？相对于竞争对手，这个产品怎么样？在狭义和广义方面，有哪些产品是这个产品的竞争对手。一套音响的竞争对手不仅是其他的音响装置，还应该包括人们所用的智能手机；
- ❑ **权衡取舍** 为了实现用户的使用目的或满足用户的需求，这个产品作了哪些取舍？为什么这个产品会有这样的优势或遇到这样的挑战？

基本上，你得站在商业角度以及用户的角度上，将这个产品解剖开来分析。要不断地问自己为什么以及如何做。为什么这个产品要这样做，如何才能做得更好？

虽然你应该假设这个产品的公司每个设计决定都是有正当理由的，但你不应该认同他们的每一个决定。

说真的，每个产品都有改进的空间。

14.6 提示与技巧

说到底，这些设计问题是为了考验你换位思考的能力。你能否站在用户的角度，考虑他们的

需求？或者说，你会不会只是设计了一款自己想要的产品？关键是要着眼于用户。

此外，以下提示与技巧会对你有所帮助。

- **有意见、有态度** 发掘一个观点，把自己当成产品的主人。面试官会看好那些有自己想法的产品经理。
- **让面试官拍案叫绝** 试着想出至少一个能让面试官拍案叫绝的主意。想到了，就说出来。你是否想到了这个产品能轻易获得一席之地的大型市场？你是否想到了一个绝妙的功能点，能让用户愿意花双倍的时间来使用这个产品？不要在考虑快速修复产品时，把这些好主意给埋没了。
- **使用白板作记录** 要是遇到这类问题，不要觉得非得死死坐在座位上不可。说实话，站起来并使用白板来解题才对，否则为什么会在面试地点放置一个白板呢？使用白板可能会令你更清晰地和面试官沟通你的设计思路。
- **不要过度设计** 有些人的设计过了火，用一些古怪的设计来解决一个简单的问题。你得现实一点，要知道什么能做，什么不能做，还要懂得取舍。
- **不要抱怨手头资料不够** 是的，我明白，在理想的世界里，你会想要先做大量的研究，收集到一大堆数据再设计。但此时是不现实的，这可是一场面试，也是现实世界。你得凑合着来，有什么就用什么。你可以简单地和面试官提及，需要哪些研究数据来支撑你的设计决定。如果面试官看起来挺感兴趣，那就不妨大胆一试，问他要资料。否则就不要纠结于此。用你最好的判断力来支持你的设计决定吧，就如在现实世界里那样。
- **考虑商业因素** 对一个产品来说，消费者的使用目的重要得不可思议，但商业目的也同样重要。如果你要批评一个真实的产品，想想它的商业目的。举个例子，如果你正在讨论如何改进一个社交媒体产品，就可以考虑商业上的重大问题。这个产品是否提高了公司的收入？是否提高了用户注册率？是否提高了用户粘性？这些商业目标也许可以引导你去改进这个产品。
- **作权衡取舍时采取开放的态度** 有些候选人喜欢鱼目混珠，以为这样做，面试官就抓不到产品的潜在问题。想得太美了。面试官会轻易看到你的方案有哪些问题。如果她确实看不出来，那你就将她不知道的问题指出来，这样还能给她留下好印象。

有疑问时，就按你想象中出色的产品经理那样去表现吧。一名出色的产品经理会对局限性表现得很现实。他会收集研究数据，但是也会在没有研究数据的情况下继续前进。同时，一名出色的产品经理会考虑商业因素，也会对设计中的潜在问题保持开放的心态。

像一名出色的产品经理那样表现吧！

14.7　样题

- 如何为孩子设计一个书柜？
- 如何为坐轮椅的人设计一个烤箱？
- 谷歌地图要发行一个供学校使用的地图版本。如果要你设计，你会如何设计这个版本？
- 你最喜欢的商业工具是什么？为什么？
- 如何设计一个社区公园？
- 如何改造一个超市，以使之更适合大学生？
- 你最喜欢的图片存储网站是什么？如果要你改进它，会如何修改？
- 设计一个可以代替 Google.com 的门户或交互式登录页面。
- 如何为企业家设计一个社交/职业网站？
- 选择一个你觉得不满意 Amazon.com 服务的目标用户群。你会如何重新设计 Amazon.com，以吸引这一个用户群？

第 15 章 案例分析问题

常有人说，产品经理就是“产品的 CEO”。不幸的是，这种流行的说法在这里多少有点不准确，你可不是产品的 CEO。

因为你既不搞收购，也不搞兼并。你不用和股东打交道，甚至不用管财政方面的事务。

然而，你的责任是将一个最好的产品交付到消费者手中。注意，这个句子有三个重要的词：“产品”、“交付”和“消费者”。绝大多数产品经理面试题都是围绕这三个关键词展开的。

15.1 案例分析问题：顾问与产品经理

如果你曾经面试过管理咨询顾问的工作，那么就应该知道，咨询案例和产品经理案例面试问题是类似的——都很有可能给你设下陷阱。

诸如麦肯锡、贝恩、BCG 等管理咨询公司的顾问职责和产品经理是很不同的。一般来说，顾问会用大局的眼界来看待公司，花费几个星期或几个月来收集数据，然后解决类似“我们的客户该不该收购____”的大局难题。他们从公司文化兼容性、对供应商的依赖性和税制结构之类的角度来看待事物。他们应该偏重于数据，应该有条不紊地办事：提出问题、收集数据、诊断问题、提出解决方案（顾问是不参与执行的）。

而产品经理关注的则是产品和消费者。他们作出产品功能和市场准入的决定，然后跟进解决方案的执行，交付并维护产品。

尽管面试问题可能会有交集（“谷歌是否应该推出电视服务”），但在两种面试中，“适当表现”的评判标准却是不同的。顾问会要求面试官提供用于解决问题的数据（或者面试官的问题直接给出了数据），而面试官更期待产品经理依赖他们的直觉来处理问题。比如说，一个产品经理面试官不会轻易地为面试者提供幻灯片。这并不是说，产品经理就绝对不能依赖数据，或顾问就不应该依赖直觉。只是，他们各自的关注点是有偏好的。

如果你搞混了，顾问会这样和你说：我可早就警告过你了！

15.2 面试官要的是什么

有一位产品经理候选人曾在面试之前寻求过我的帮助，在最近一次顶级创业公司面试中，他被问道，如果他是公司的 CEO，会做些什么。“她怎能问我这个问题呢？”迈克尔抱怨道，“我告诉她，这个问题对我来说很不公平，因为我对这家公司的了解显然没有她那么深。”可以说，那场面试并不顺利。

迈克尔们，你们听好了：你当然没有真正解决这个问题的知识！如果你有那些知识，当然可以将问题解决得更好！但这不是重点；说实话，这甚至不是问题。

面试官并不是根据你给出的答案的精确度来给你打分的，他是要通过你解决问题的过程来考验你的能力。要达到这个目的，你就不需要现实场景中才有的那些数据和背景资料了。

面试官要找的是拥有这些能力的候选人。

- **将问题结构化** 哪怕是一些看起来开放回答的问题，也可以（应该）按步骤拆解。市场可以细分，策略也可以拆解。想个结构化的办法，将问题攻下。
- **表现出强烈的直觉** 一般我们手头上都没有需要的所有数据，正如我们也不知道未来会如何一样。一名出色的产品经理应当具有良好的商业决策能力，哪怕手上没有详尽的数据也是一样。
- **掌握主动权，而不要只看着办** 你可能不是产品的 CEO，但你就是产品的领导者。你得通过主动推进这场面试，来展现你的领导才能。回答问题要相对详尽（讨论产品的利益和权衡取舍，短期和长期的好处，等等），并能够提供依据来支撑你的答案。

不过，也不要表现得太过火了。如果你发现面试官追问了很多问题，而你每次都只是给出简短的回应，那你在面试中的扮演的就是走着瞧、看着办的配角，而不是主动驱动的角色了；另一方面，如果你担心自己实在太唠叨，那就询问面试官是否需要展开话题。你可以（也应该）询问面试官，但也要有个限度。如果面试官把球踢回给你（比如他这样回应道：“那么，你认为呢？”），那就说明你确实是在这个问题上扯远了。没关系，再回到正题：想想这个问题，作出最出色的决策，然后说明你的依据。

15.3 有用的分析模式

绝大多数产品经理的职位是不需要 MBA（工商管理硕士）学位的，因此要解决问题，也多数是不需要商业模式的。毕竟，很多面试官也不知道这些商业模式。

但是，话说回来，懂得这些模式可能会对面试有所帮助。这些模式可以帮你将问题拆分为清晰的步骤，或为你指出可能被你大意遗漏的地方。只要别冷不丁地甩出这些模式，然后非把问题套进去不可就行了。结构化的思路很重要，但风马牛不相及可就是另一回事了。

不用担心记不住这些模式；你不用知道这些模式。只要阅读一遍，让自己有个底，知道有这么一些可以套用的结构类型就可以了。所谓的分析模式，只不过是一种将大型、复杂的问题分解为小问题的结构。遇到案例分析问题，你应该考虑如何将其打散、分解为小问题。这些模式提供了分解问题的示例方法。

15.3.1 消费者购买决策过程

有很多模式能表示消费者作购买决策的过程，其中以 AIDA 和 REAN 最为常见。

AIDA 模式将消费者的购物决策划分为注意力→兴趣→欲望→行动。

- **注意力** 你需要以某种方式得到消费者的注意。如何设计一个能抓住他人眼球的电子邮件标题？一则时髦的广告呢？或者由一个靠得住的朋友或者网站来推荐？
- **兴趣** 既然已经得到了消费者的注意，就需要让他们对你的产品感兴趣。那么，你的产品有什么优势或者好处呢？
- **欲望** 随着消费者的兴趣被激起，你要让他们相信，他们想要你的产品。
- **行为** 最后，消费者想要得到你的产品，他们就会购买你的产品了。

REAN 模式将 AIDA 扩展开，加入了购后行为。

- **触及** 消费者知道了你的产品。
- **兴趣** 消费者对你的产品感兴趣，并考虑购买。
- **行动** 消费者采取行动，购买了产品。
- **培养** 消费者已经购买了产品，现在你的责任是经营你与消费者之间的关系。

不知道使用哪一个模式？没关系。再强调一次，特定的分析模式并不重要。关键是要明白，购买行为是一个周期，消费者要先知道你的产品，然后将产品与他的需求及竞争对手的产品进行对比评估，最终才买下来。消费者做出购买决策后，客户关系就延续下去了，此时你的公司就应当下工夫来经营这段稳固的客户关系。

在讨论如何营销一款新产品时，这个模式应该会有帮助。你可以谈到，要获得消费者的注意相对容易，但是“行为”部分（让用户放弃我们的竞争对手，转而使用我们的产品）就比较困难。

15.3.2 市场营销组合（4 个P）

市场营销组合（也称“4 个 P”）是理解产品营销几个方面的一种方法。

- **产品（Product）** 当然，这里指的是实际售出的产品。它应当迎合消费者的想法或需求。
- **定价（Price）** 产品的价格将决定消费者购买产品的数量以及消费者的类型。相比起实物产品，线上产品和服务的价格策略更加复杂。比如说，一个线上存储服务可以提供首月免费试用，后续服务可提供折扣给非营利用户，而自动备份工具则作为附加功能，下单才收费。
- **推广（Promotion）** 推广包含了所有形式的广告、公关、口碑营销以及人员销售。比如说，一款年轻人使用的产品推广可能会包含分发免费赠品给有影响力的博主等。
- **渠道（Place）** 一款实物产品的发布可能会包含亚马逊线上推广、像苹果公司那样开商店、在零售店开发布会以及在自营网站作销售活动等。并不是说发布会越大就越好；很多公司更愿意以限制销售渠道来把握他们的销售经验。对于线上产品来说，“发布”可能只是一次网上发布，也可能是将其与公司其他产品捆绑起来出售。

对于线上产品来说，推广可能就变得非常复杂。许多产品都在为得到消费者的注意而竞争，而广告通常不足以推动销售。

这个模式对于讨论一款新产品或现有产品的市场营销计划的不同因素应该会有帮助。

15.3.3 SWOT分析法

SWOT 分析法（态势分析法）是一个对公司及其产品进行分析的结构。

- **优势（Strength）** 优势是能为产品带来收益的内部因素。它包含任何关于成本、产品特性、企业文化、知名度、基础架构或其他方面。举个例子，在考虑推出 Kindle 阅读器时，亚马逊的一个优势就是它本身已经是消费者在线上购书的地方。
- **劣势（Weakness）** 劣势是能体现一款产品所面临挑战的内部因素。举个例子，如果一家网络公司打算推出一台实物设备，它的一个劣势可能就是它并没有任何发布实物产品的经验。
- **机会（Opportunity）** 机会着眼于外部环境，与市场增长、技术变革、市场竞争和法律法规等因素有关。举个例子，医疗保健的成本上升为能使用户更深入了解健康状况的产品创造了机会。
- **威胁（Threat）** 威胁是一款产品所面临的外部挑战。举个例子，能源使用规定的不可预测性构成了绿色能源产品面临的威胁。

下面的表格描绘出了 SWOT 结构：

	好处	坏处
内部	优势	劣势
外部	机会	威胁

这个模式不仅能帮助公司决定它是否应当追求某个机会，还能分析什么策略可以使这个机会有更长远的发展。

15.3.4 情境分析法（5 个C）

这 5 个 C 描绘了一个产品或决策所处环境的概况。

- **公司（Company）** 这个 C 涉及了公司的所有方面，包括它的产品、文化、战略、品牌知名度、优势、劣势以及基础架构等。
- **竞争者（Competitor）** 竞争者包括直接竞争对手、潜在竞争对手和替代产品。针对每一种竞争的讨论，应当涉及市场份额、产品设计的权衡取舍、产品的市场定位、产品的使命和产品未来走向的决策。
- **客户（Customer）** 这个 C 包括人口统计、购买行为、市场大小、分销渠道以及用户需求。
- **合作者（Collaborator）** 合作伙伴包括供应商、分销商以及合伙人。关于合作伙伴的讨论，可以涉及到赋予某合作伙伴以价值的特点，以及如何促使产品成功。
- **外部环境（Climate）** 外部环境包括法律法规、技术变革、经济环境和文化趋向等。一个敌对的外部环境可能会扼杀一个商业决策，而一个友好的外部环境则能大力促进成功。

这个模式在公司就“是否应当推出某产品”或“应采用何种策略”作决策时有引导作用。

15.3.5 波特五力模型

波特五力模型是一个对行业进行分析的模式。这个行业分析法有助于理解公司的决策。

- **行业内现有竞争者的竞争能力** 一般，竞争对手越多，竞争就越火热，直接竞争也就越多。如果有许多公司做出了大同小异的产品，那么每个产品的价格自然就得下调了。很多事情都能影响市场竞争，比如说市场的增长（市场增长使得市场竞争者不必与他人争夺市场份额即可实现自身的扩展）和退出市场的高成本（这使得众多公司不愿退出市场）。
- **购买者的议价能力** 如果一家公司或一个行业的购买者相对稀少（比如说购买者只限于政府或大型银行），或者一些购买者的购买力占公司或行业的收入特别大，那么这些购买者就有相当大的议价能力。这个能力允许他们影响产品的定价，产品的功能，产品交付的时间和其他方面等。
- **供应商的议价能力** 供应商和购买者一样，如果公司严重依赖供应商，那他们也能给公司造成影响。一般来说，只有当某零件必须从独一（或极少）的供应商取货时，这种情况才会发生。
- **替代品的替代能力** 竞争并不仅仅直接来自竞争者，也有来自替代品的竞争。举个例子，

虽然亚马逊是电子书的唯一卖家（因此并无直接竞争者），但电子书的价格仍受到纸质书竞争力的影响。

- **潜在竞争者进入的能力**　如果进入一个行业几乎没有门槛，那么该行业的公司就常常会受到竞争的威胁。如果他们的产品定价太高，任意一家公司都能进入市场抢占它的市场份额。行业的门槛可能包括专利技术、大型规模经济、强势品牌或任何较难实现的事情。

举个例子，就说 PC 电脑市场吧。由于很多销售额都来自小部分的零售商，购买者就有了相当可观的议价能力。另外，由于某些零件只有特定的制造商才供应，而另找供应商的成本又太高，因此供应商也有相当大的议价能力。从好的方面看，竞争者和有限的替代品还是有区别的。要进入市场，有一些不高不低的门槛（如品牌），有很多糟糕的市场竞争品，也有很多更好的市场竞争品。

在讨论公司是否应进入某特定市场时，这个模式可以派上用场。这个行业怎么样？如果它是一个高度竞争的行业，我们可能会选择避开它。

15.3.6　打造你自己的分析模式

如果你觉得这些模式看起来都差不多，没关系，并不是你一个人这样觉得。

它们确实有交集，但下面也可以看到它们的区别。

- **消费者购买决策过程**　帮助我们理解购买过程，还提供了一个产品促销的切入口。
- **市场营销组合（4 个 P）**　描述了公司市场营销计划的 不同方面。
- **SWOT 分析法**　提供了分析企业战略决策的模式。
- **情境分析法（5 个 C）**　描绘了一个产品或决策所处环境的总览。
- **波特五力模型**　描绘的是一个行业整体的情况。

从这些模式入手来分析一个问题，并标记出可能会遗漏的关键因素，比如经营旧客户关系的价值等等。

在一场真实的面试中，你应该使用最佳“模式”或结构来解决问题。如果上述这些模式当中有适合的，那就太好了！但是，很多情况下，你需要创造自己的分析模式。你可以稍微修改上述的某一个模式，或将其中两个模式融合起来，甚至是想一个完全新奇的模式。

15.4　产品衡量标准

关于产品衡量标准和公司指标的讨论可以有很多种方式。它们可能会在诊断某种假设性问题时出现，或者被用来解释公司作出某一决策的原因。

在讨论过程中，确保要定下对产品最重要的那一个衡量标准。任何产品都会在某方面占有优势，而在其他方面有待改进。

15.4.1 衡量标准的类型

根据不同分析目的，你可以将产品衡量标准划分为很多类型。下面，我们将按用户获取、用户活动、用户的转化与维系、金钱来划分各种衡量标准。

你也可以按客户生命周期来划分它们。如何按客户生命周期划分？戴夫·麦克卢尔（Dave McClure）在他的“海盗的创业指标”[①]演讲中提供了很好的范例：用户获取、用户激活、用户维系、用户推荐、收入。

1. 用户获取

- ❑ 我们有多少用户？
- ❑ 用户基数定期增长率是多少？为什么？
- ❑ 我们有多少活跃用户？如何定义活跃用户？
- ❑ 用户从哪里来？用户会推荐他们的朋友使用我们的产品吗？
- ❑ 哪些渠道最利于获取用户？

2. 用户活动

- ❑ 有多少用户使用某功能？
- ❑ 有多少比例的用户完成过某个完整工作流？
- ❑ 人们对这个产品有什么看法？他们喜欢这个产品吗？你能对这方面进行量化吗？

3. 用户的转化与维系

- ❑ 用户转化率是多少？（用户转化即免费用户转为付费用户、游客转为注册用户等等）
- ❑ 用户流失率是多少？

4. 资金问题

- ❑ 用户获取的成本是多少？
- ❑ 为一名客户提供技术支持的成本是多少？
- ❑ 一个用户可带来多少收入？（即每用户所带来的平均收入）
- ❑ 客户的终身价值是什么？
- ❑ 我们的收入增长率是多少？

15

① “Startup Metrics for Pirates.” Dave McClure. 8 August, 2007. http://www.slideshare.net/dmc500hats/startup-metrics-for-pirates-long-version

15.4.2 测量

一般说来，在一个测量标准中，应该测量其变量，而不是总量。测量这项任务就是一项消耗时间的工作。

这些测量值一般是公司用来展示给投资者或者媒体看的，它们并不能帮助公司作决策。

很多衡量标准可以（也应当）划分为同类用户群。这些用户群可以依据性别、地域、注册日期或很多其他因素来分类。

要收集有用的数据，你有很多选择。

- **可用性测试** 这个方法一般不会提供实际的测量值，但它可以帮你理解出现各种值的原因。比如说，为什么用户会在某个节点上离开你的站点？
- **用户反馈** 用户反馈可以来源于社交网站、客户服务页面或客户调查。这个方法在分析测量值时能提供场景，它和可用性测试一样，在分析方面比收集数据更为强劲。
- **流量分析** “谷歌分析”等工具可以帮助公司了解用户与网站互动的方式。
- **内部日志** 比起流量分析，直接分析登录日志信息可以帮公司更加深入地了解用户行为。
- **A/B 测试** A/B 测试可以通过对使用某功能的用户群行为和不使用该功能的用户进行对比，来帮助公司了解一个特殊的功能改动对全局带来的影响。虽然 A/B 测试是一个极其好用的工具，但它还是可能会被误用。举个例子，为一小部分用户推出新的聊天功能，可能会产生一些关于用户使用模式的误导性数据。毕竟，要是我的朋友都不能使用聊天功能，我跟谁聊天呢？

15.5 面试问题

并不是所有面试问题都能完美归入以下分类，但还是有很多问题适用这些分类，或者说，至少能对得上几个分类。以下是案例分析问题的主要类型，以及套用这些类型的方法。这些类型包括战略问题、市场营销问题、产品发布问题、头脑风暴、定价问题以及处理难题等问题。

要解决每一类问题，一个有条理的方法至关重要。想办法将一个问题拆解成若干小问题，然后逐个击破。

15.5.1 战略问题

战略问题有两种，一种是问你某公司的战略是什么，一种是如何制定战略。要点是，你得在两个层面上都偏重考虑产品的战略。

- **微观层面** 这款产品的商业模式是什么？ 公司为该商业模式的成功，做了哪些努力？消费者会想要这些吗？

- **宏观层面** 这款产品是否符合公司的企业愿景？它是否会带来新的发展机会？它是否有把握进入现有市场？

很多产品在两个层面上都有备而来。举个例子，亚马逊的 Kindle 阅读器收入独立于亚马逊，但由于购买阅读内容必须去亚马逊，它也巩固了亚马逊的市场地位。

探讨微观与宏观战略时，SWOT 分析法和波特五力模型等模式可能会派上很大用场。要考虑以下问题：

- 公司的使命是什么？
- 公司的目标是什么？
- 公司的优势是什么？
- 公司的劣势是什么？
- 公司如何利用自己的优势，或如何弱化自己的劣势？

在微观和宏观层面上考虑这些问题，然后将公司目前的做法进行对比分析，就能借此描述公司的战略了。

待评估的潜在战略可包括以下几方面。

- **实现多元化收入来源** 公司最好还是要实现收入来源多元化，否则市场的剧变或新竞争者的崛起就可能会一招致命。
- **建立竞争者壁垒** 建立门槛能将竞争者拒之门外，也由此保证了公司自身的收入。举个例子，Facebook 对社交网站的绝对优势使得其他竞争者在事件管理工具类产品的竞争上困难重重（但也不是完全不可能成功）。
- **成为某物的“一站式商店”** 公司还可以在特定领域中拓展自己的产品配件，这能成为竞争者的壁垒。举个例子，由于产品量特别高，亚马逊被看作购物必去的地方。而在某领域独占鳌头的公司某种程度上可以对亚马逊的这一地位造成威胁，这使得它们成为亚马逊的并购目标。
- **成为低价的行业领袖** 定价便宜会削减利润空间，但也能使得现有或潜在的竞争者望而却步。注意，主动让自己的价格变成行业最低与为获得竞争力而被迫降价是不一样的。前者是积极主动的战略决策，而后者只是一个结果。
- **减少对重点消费者或供应商的依赖** 过度依赖单一的消费者，你可能会被迫满足他们的需求；过度依赖供应商则可能造成出货意外延迟或质量下降。
- **对新市场进行试水** 有些时候，公司会将一个利基产品投入新市场进行试水，建立自己的品牌并获得这个领域的更多信息，从而打开该市场。然后，“真正的”产品再迎头赶上。

这个清单只为提供参考，让你知道哪些战略可以成为公司最终的决策。除了这些，还有大量的战略可以作为参考。在准备面试的时候，将你见到的战略记下来会对面试有所帮助。

1. 例题分析

Facebook 以 10 亿美元收购了 Instagram，尽管 Instagram 根本没有收益。你认为 Facebook 为什么要这样做？

回答这个问题，并不需要对 Instagram 有太多了解。面试官在乎的并不是这个。一个对 Instagram 不甚了解的人也能给出一个强有力的答案，就像这样：

> “嗯，这个问题有意思。说实话，我自己很少使用 Instagram，但我对它有点印象。至于 Facebook，我当然就很熟悉了。我想先将这个问题进行分解，先分别探讨构成这次收购的各个关键节点，然后再分析这些部分如何与 Facebook 的企业使命相一致或相悖。
>
> Facebook 的企业使命是把人们联系起来，并帮人们分享他们的生活。通过这次收购，他们获取了三样东西：公司（雇员）、产品、用户。我们从 Facebook 的角度来考虑这三样东西。
>
> Instagram 肯定有一些很优秀的雇员，但当时他们的公司还相当小。我无法想象这是一个以 10 亿美元收购公司的强劲理由。
>
> 他们的产品挺有意思的。Instagram 创造了一个主打照片分享的优秀产品。大概就是这一点引起了 Facebook 的恐慌。对于一家将使命定位于帮助人们分享自己生活的公司来说，分享照片这一功能无疑是至关重要的；这个功能是 Facebook 的一大杀手锏，也是一个关键的优势。但是无端出现了另一家公司，在这方面也非常强大，而且还在 Facebook 的老本行上给它以痛击（至少在某些方面）。要是人们越来越经常使用 Instagram 而不是 Facebook 来分享照片，会怎么样呢？这是一个很大的威胁。
>
> 用户是 Facebook 收获的另一个重要部分。我记得，Instagram 被收购的时候拥有大量用户，而且已经建立起了他们自己的社交网络。显然，这就是 Facebook 的老本行，而且 Instagram 在这一点上也大获成功。要是他们的社交网站壮大了，会怎么样呢？是否会有其他公司（尤其是一些创意网站，如 Etsy）开始选择和 Instagram 而不是 Facebook 合作？Facebook 所持有的用户基数正是其他竞争者进入市场的高门槛，而每崛起一个社交网络，Facebook 就被迫将市场开放一部分。
>
> 归根结底，看起来 Facebook 大概是发现有那么一个年轻的创业公司，突然间在 Facebook 最擅长的两个领域中崛起：照片分享和社交社区。Facebook 大概觉得他们不能在这两个领域冒险，或者说他们不能容忍一山二虎的存在。此外，他们可能还对 Instagram 的下一步行动感到过恐惧，或者说，他们可能也担心，万一谷歌抢先一步把 Instagram 买下来，后果可能就不堪设想了。

这位候选人可能还遗漏了 Facebook 收获的一些方面（比如说 Facebook 得到了占领移动应用市场的机会），但对于一个不熟悉 Instagram 或 Facebook 的移动应用战略的人来说，这是一个可以理解的疏忽。

注意看这位候选人是如何梳理思路的，她先将 Facebook 在收购中的收获分解，然后将 SWOT 分析法融入分析，她提到了优势、劣势和威胁。至于我们先前所提到的分析模式，这就是一种应用方法。

然后，她将答案总结起来，给出了一个清晰、简洁的结论。

2. 样题

- 请解释谷歌进入手机市场这个决策背后的战略。
- 亚马逊旗下有很多独立网站，它们或多或少都和亚马逊网站有重复的功能：Zappos（鞋子、衣服）、Diapers.com（婴儿及儿童用品）、YoYo.com（儿童玩具）、Look.com（儿童服饰）、Soap.com（化妆用品）、Casa.com（家居用品）。亚马逊要维护这么多功能类似的网站，他们的战略是什么？你认同这个战略吗？
- 如果你是亚马逊的决策人，你会在印度推出服务吗？为什么？如果是你，你会如何做？
- 亚马逊在自己的网站上挂有在其他网站能买到的产品的广告，你认为他们的战略是什么？这是不是一个好的战略？
- 你认为苹果商店是否应该出售非苹果产品？为什么？
- 如果你是微软手机的负责人，你会对它做什么决策？
- 谷歌有哪些你不欣赏的产品或服务？为什么？
- 如果你是雅虎的 CEO，你的战略会是什么？
- 想象一下，要启动一个社交网络服务。你的战略会是什么？
- 想象一下，你正在考虑推出两个收入和成本都差不多的服务，你要如何决定推出哪一个？

15.5.2 市场营销类问题

市场营销问题一般与产品定位、客户和处理竞争关系有关。市场营销组合（4 个 P）解决这类问题尤其奏效，它能提供一些产品内容相关的想法。SWOT 分析法、5 个 C 和消费者购买决策过程的视角也可以派上用场。

要攻下这类问题，可以尝试按以下方法来解决。

(1) 了解公司：公司的目标是什么？公司的使命是什么？公司的优势和劣势是什么？它面临着什么威胁？

(2) 了解市场竞争：将市场竞争进行细分。哪些竞争者在什么市场与我们竞争？他们对自己如何定位？他们如何巩固自己的市场地位？他们的优势和劣势是什么？

(3) 了解消费者：我们的用户是什么人？这款产品能满足用户的什么欲望或需求？用户的购买行为和使用行为如何（如一次性使用或重复购买等）？

(4) 了解市场全景：我们的产品是否存在法律争议？如果产品只针对某个地区，这个地区有

什么特点（比如说，频繁电力中断）？有什么外力能决定你的市场份额？

(5) 推销你的产品：根据上述几个步骤中谈论的信息，针对一些诸如产品及其定位，或者你的推销策略（此处可以套用消费者购买决策过程模式）之类的问题作出一些决策。根据面试官或问题的要求，可能你还要探讨产品定价和产品发布的问题。

注意，应付这类问题，要大量涉及到公司或产品的背景信息。这是正常的；在讨论之后，你还应该提出一个市场营销计划。

1. 例题分析

如何向开发者推销 Windows Phone 以鼓励他们接受这个开发平台？

嗯，首先我要先想想目前的市场是怎样一个局面，包括公司情况、竞争情况、消费者、市场全景，等等。

- ❑ 就我所知，微软关于开发者的目标应该是要增加高质量手机应用的数量。目前 Windows Phone 所占的市场份额还相当小，这就是该目标的壁垒，并导致了另一种后果：Windows Phone 平台并不吸引开发者。而从另一方面看，它也有优势，Windows 系统在台式机市场占有主导地位。这就意味着至少在某种程度上，它易于开发，还附带高知名度品牌。这就是微软可以利用的优势。
- ❑ Windows 手机面临的两个最大的竞争者是 iPhone 和 Android 手机。iPhone 在高端市场独占鳌头，引得开发者对 iOS 平台趋之若鹜。一般认为，iPhone 用户下载和购买手机应用的花费比其他平台的用户更多。Android 系统吸引开发者的是它的市场份额，但它有大量设备需要支持，这种兼容性问题可能会令开发者沮丧。
- ❑ 在这个案例中，我们的用户是应用开发者。不过，开发者的决定多少也要依赖于用户需求。开发者有很多类型：业余爱好者、企业开发者、创业开发者，等等。开发者想要的是一门能快速而简单地建立一款产品的开发语言，还需要平台有能力将他们的产品推出给尽可能多的用户。
- ❑ 最后，在总结出一个可行的市场营销计划之前，还需要对市场环境进行探讨。我们需要考虑，当下有什么时势可能会打开市场的缺口？其中一个就是黑莓手机的陨落给企业市场造成的相对空白。iPhone 和 Android 手机多少填补了这个空白，但可能还有许多空间可待填补（如果你能得到 IT/操作系统管理者的支持，更是如此）。另一个与市场环境相关的部分，就是因国家不同而造成手机的多样化。我不知道这方面的统计数据，但我猜想应该还有一些 iPhone 和 Android 手机并不占主宰地位的市场。

现在，我们说说市场营销计划。我认为主要应该专注于以下几点：(1) 使得 Windows 平台的开发尽可能无障碍；(2) 确保 iPhone 和 Android 上的精品手机应用也有 Windows 版本；(3) 找到市场缺口。我们针对开发者的市场定位是：要使这个平台易于开发、维护和出售手机应用。

- 就使平台易于开发而言，我们已经占了上风，因为很多开发者已经懂得 Windows 开发了。我们需要为 Windows 手机开发者创建一套优秀的教程和示例代码。如果可能的话，也可以尝试开放一些优秀的 API 供开发者植入他们的应用，以便他们建立关键功能，如即插即用编码等。
- 此外，开发者可能会在网络上寻求帮助。对此，我们要建立起一个 Windows 手机开发主题社区，并提供免费的开发技术支持。
- 同时，我门要找到 iPhone 和 Android 手机在开发环境上的弱点。我本人没有做过足够的手机开发，但是我听说 iPhone 开发者经常抱怨应用下载量的来源资料不足，我们可以将这一点作为突破点。
- 确保 iPhone 和 Android 的精品手机应用也有 Windows Phone 版本，这一点我们采取主动。我们需要主动去追踪两个平台上流行的手机应用。如果这些团队还没有 Windows Phone 手机，那就免费给他们送几台。如果是个人开发者，而不是整个公司或创业团队，我们甚至可以帮他们测试，提供技术支持，以使手机应用的质量达标。
- 最大的市场缺口大概是在企业领域，这也是我们要专注的地方。我们可以争取企业开发者和 IT 人士来专攻 Windows Phone 开发和技术支持。他们当中很多人就是做 Windows 开发的，所以这对他们来说并不是什么大问题。如果我们有更好的工具可以为企业雇员提供支持，就能鼓励他们在选择雇员工作手机时选择 Windows Phone 手机而不是 iPhone 和 Android 手机。

这个营销计划是为了告诉开发者，我们有全套生态系统工具：教程、API、技术支持、免费设备、社区、分析数据、管理工具，等等。只有将开发者的利益放在首位，我们才能获得他们的青睐，从而使得我们成为最容易开发和支持的手机平台。

这些不一定都是最好的，也不是唯一可接受的市场计划。对于这个问题，还有很多市场营销计划是可行的，哪怕是一些结论完全相反的计划也可以。

毕竟在面试时，应聘者是脱离了数据来制定营销计划的，所以只能指望你知道的（或你认为的）事情是对的。如果你对某些事的理解有误，就可能会提出一个面试官知道肯定会失败的建议。

一名好的面试官可能会在你出现理解有误时给你一些提点（因为行业知识对解题可能很重要），但如果没有，他也应该在评估你的表现时，先假定你的理解是对的。

记住，这些问题的目的是考验你在几个方面的能力：对开放问题应用结构化解题思路、作英明的决策、考虑产品的市场营销计划。哪怕你手上只有“不利”的信息，也可以将这些事情做得很好。

2. 样题

- 你认为营销得很好的一个产品是什么？你会如何使它变得更好？

- 如何营销 Gmail?
- 如何营销安卓平板电脑?
- 你认为谷歌为它的社交产品所作的市场营销怎么样?你会如何改进他们的计划?
- 如何营销 Windows 系统的下一个版本?
- 要扩展亚马逊的网络服务,你会如何选择目标市场?
- 你认为现在的全天然食品应该如何营销。哪种营销方式效果好?如何才可以使它们更好?
- 如何营销一个能够追踪体重、卡路里消耗和体育锻炼的手机应用?
- 9. 你会如何在学校里营销 Google Docs?
- 10. 你如何营销 Kindle 阅读器的一个杂志订阅服务?

15.5.3 产品发布类问题

发布产品是产品经理最重要的日常工作职责之一。所以在面试上,面试官难免会问你如何发布一个产品。

要解决这类问题,以下方法可能会派上用场。

(1) *产品*:谈谈产品的愿景,以及产品的优势、劣势和风险。这款产品最终希望达到什么目标?它的目标市场是什么?你对这款产品有什么担忧?这款产品的竞争者是谁?这款产品在竞争中的地位如何?

(2) *产品的发布目的*:确定产品的发布目的。想要尽可能多的用户?想要确保产品收益?想要确认产品是否符合市场需求?想要看到市场出现积极反响(哪怕要以减缓收益增长速度为代价)?

(3) *产品发布部署*:确定了产品发布的目的,就要决定如何达到目的。什么市场可作为合适的测试平台?是否要通过邀请系统来控制用户量?你是否会为了提早发布,只推出一个限制功能的版本?你会不会试着让这次发布尽可能地引起轰动,从而获得尽可能多的用户?

(4) *产品发布(发布准备+发布+发布后)*:基本的部署定下来后,你需要考虑如何落实产品的发布。将你的产品发布分为三个阶段:发布准备、发布和发布后。产品发布可不是一天两天的事。

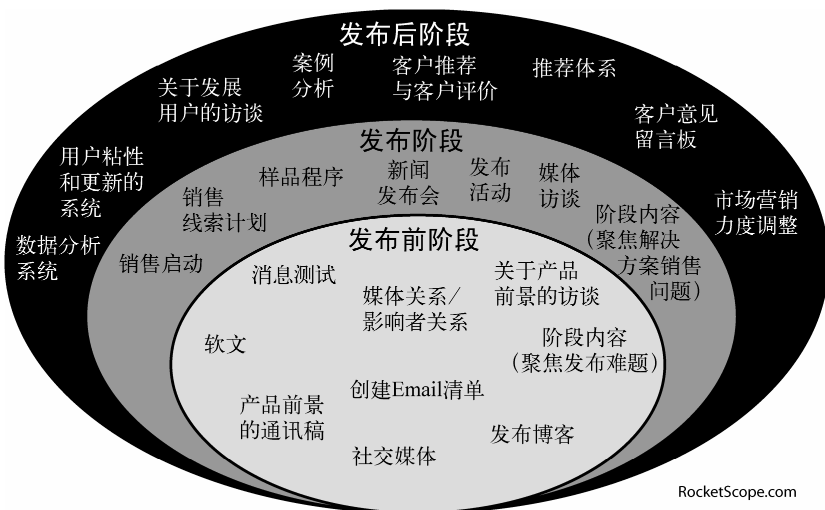
关于产品发布(包括三个阶段)的讨论,应该涉及以下方面。

- **目标市场** 你是否会将产品先发布给指定城市或学校?发布初期的目标市场是什么?
- **用户类型/用户构成** 如果有多种类型的用户(如司机和乘客),我们可能会针对用户类型分别进行发布:不同的发布时间、不同的发布方式。
- **MVP 还是全款产品?** 你会先发布一个最小可行产品版本,还是等到这款产品的功能完全成熟了才发布?
- **分销** 你会在什么商店或销售渠道出售这款产品?你会在什么平台发布这款产品?
- **推出** 这款产品何时会发布给不同类型的用户?你是否会先发行一个内测版本?是否需

要邀请才能参加内测？

- **制造新闻** 你会如何为这款产品制造消息（或者说，你会否为它制造消息）？你会找哪些人帮你发布消息？如何获得他们的认可？你希望他们为这款产品做哪些宣传？
- **合作伙伴** 在这款产品上，有没有合作伙伴会帮助你实现你的愿景？
- **风险** 这款产品面临什么风险？你会如何规避它们？这款产品是否面临法律挑战？你是否担心，用户量会因隐私政策的调整发生反弹？要考虑这些决策问题，消费者购买决策过程可能会派上用场。哪些部分是最困难的？你会如何克服它们？

一些可行的产品发布活动，都在下方的图示中列出来了①。



1. 例题分析

假如你正在为本地服务供应商（如管道工人等）创建一个用于与用户联系的推荐/预约服务，一个类似开放式表格的服务。你会如何发布这款产品？

候选人可以这么回答：

> 好的，首先我们要明确：这是一款可以用来搜索管道工人、电工等，还可以查看评论的产品，大概还可以预约工人的时间空档（如果可以的话），或者发布一项工作来招标。
>
> 我认为这款产品在满足用户需求方面十分清晰。在这个需求上，我没有见过任何直接竞争者。
>
> 我们将面临一个挑战，这也是产品发布部署需要考虑的问题之一：一个“先有鸡蛋

① “Launching a Startup? Plan Your Marketing Around These 3 Phases.” Dunford, April. 22 July 2013. From http://www.betakit.com/launching-a-startup-planyour- marketing-around-these-3-phases/

还是先有鸡”的问题。

一般而言，我们有两个类型的用户：服务供应商和客户。服务供应商不会加入我们，除非有足够多的客户在使用我们的产品。而客户也不愿加入，除非我们有足够多的服务供应商。

另一个挑战就是如何在用户需要服务供应商时，成为他们的首选产品。这就需要我们尝试改变用户行为，这通常很难。

我们要通过部署产品发布，来处理这两个问题。实际上，考虑到我们面临的挑战以及产品初次发布的出错几率，我想，还是先发布一个试用版本为好。

要解决第一个问题，我们要在一个专门的地区先建立一个强大的服务供应商用户库（最理想的是某个屋主众多的富裕地区）。要确保这场发布的可控制性，如果服务供应商数量太高导致大量用户涌入，就需要事先控制用户量。

要解决第二个问题，就需要让更多用户在需要的时候想到我们。最好的办法大概就是专注于用户较常用的服务：房屋清洁工、司机、园丁、泳池清洁工，等等。

同时，我认为至少要为服务供应商发布一个 Web 客户端和移动客户端，客户方面亦然。服务供应商需要两个客户端，是因为手机客户端可以让他们在工作的路上接受预约，而 Web 客户端则可以支持那些没有手机的服务供应商。对客户来说，我会建议，先从 Web 客户端入手。我确实认为手机客户端非常重要，但这样一来，我们就需要为多个手机操作系统建立手机客户端。由于我们需要尽快做出这款产品，我认为最好先做出 Web 端，目前一个就好。

产品发布的前几个月里，我们会先吸引服务供应商来注册。相比客户而言，服务供应商不会那么容易因为供需不匹配而弃用这款产品。

我们会从两个地方拉拢服务供应商。第一，从评论网站（如 Yelp）和社区博客或论坛拉获用户；第二，我们会找房地产经纪人合作。新的住户有什么问题一般会找他们咨询。要让他们知道我们的产品，还要拿到他们的推荐服务供应商清单。要达到这个目的，一支积极的营销团队是必不可少的。

针对客户的发布，我们仅提供邀请注册，控制产品传播。但是同时也要激励用户去邀请他们的朋友来加入，所以可以考虑可为邀请朋友注册的用户提供奖励。

要制造新闻，就需要在当地社区下工夫。可以在客人流量多的咖啡厅设置一个招牌，在本地博客页面或 Facebook 上关于当地社区的群组来宣传造势，等等。很多地区都有他们自己的报纸，我们也可以在这方面进行尝试。

到了发布后的阶段，就需要对用户注册量、用户转换量等衡量标准进行分析。我认为很多用户注册之后，到了真正有使用需求的时候，早已把我们忘记了。我们需要仔细追踪这个问题，因为这将成为我们要面临的一个大问题。

但这并不仅仅是一个数字游戏——我们需要和服务供应商建立起牢固的关系，需要他们承担这个网站的代言人角色。如果他们能鼓励现有的客户来使用这款产品，那么这些客户也可能会使用我们提供的其他服务。我们也需要供应商的反馈，以帮助完善这个

网站。

那么，概括一下，我们要做以下这些事情：为服务供应商提供移动客户端和 Web 客户端，为他们的客户提供 Web 客户端；从 Yelp、社区博客和论坛、房地产经纪人处拉拢服务供应商，尤其要注意那些需要多次提供的服务，如园丁和清洁工；在一处本地社区进行发布（硅谷应该是个理想的地方），尽可能多地获取服务供应商；以邀请注册的方式，逐步将服务推出给客户；与服务供应商、客户都保持良好的关系，这能作为我们在客户服务上胜人一筹的明证。

这样，就能确保可以收集到我们所需要的客户反馈，以完善我们的产品。

这位候选人认识到这个问题其实包括两次产品发布：为客户发布、为供应商发布。然后，他将两个方面的市场发布都描述了一遍，讨论了可能会面临的问题。最后，他以一个总体方案的概括做了总结。

2. 样题

- 说说你认为发布得很成功的一款产品。为什么？
- 说说你认为发布得很失败，但本身很出色的一款产品？如果要你来部署发布，你会如何做？
- 你会如何推出谷歌自动驾驶汽车？
- 假设雅虎打算推出一部手机，你会如何发布它？
- 你会如何发布 Amazon.com 的一个电子商店？
- 谷歌打算推出一款专为企业定制的 Google Docs 产品，你会如何对其进行发布？
- 对于一家帮助用户创建优秀在线广告的公司，你会如何发布公司的这个产品？
- 你设计了一款出众的水壶：它能为饮料保温，且坚固密封。你会如何发布这款产品？
- 你开发了一辆使用新燃料的车。这款新燃料非常便宜，但效果和标准燃料一样好。你会如何发布这款车？
- 你会如何发布一个印度的日用品派送服务？

15.5.4 头脑风暴类问题

头脑风暴类问题通常和创意有关。面试官想要看到，你并不是一个墨守成规的人。他们想知道你能够想出一些大胆、绝妙的主意。

对某些人来说，这并不是件容易的事。他们害怕提出这些想法会被认为很愚蠢，或者因为太紧张，想事情都是一根筋，不会转弯。

如果你被问了一道头脑风暴类的问题，试着这样做。

(1) 抛开疑虑。尽管说一些愚蠢的想法，没有关系，因为所谓“愚蠢”的主意经常是那些能让面试官为之振奋的主意！如果被“这种想法好意思提出来吗”这种问题所困扰，你的创意往往就被扼杀了。尽情释放你的想象力吧！

(2) 考虑产品优势和主要优点。如果你觉得自己技穷了，没法再想出更多好主意了，那就试着先别想这款产品，转而想想它的优势和主要功能吧。比如说购物车吧。它的优势就是它能装得下很多东西，而且它相对尺寸比较小（购物车能互相“堆叠”起来，因此所占空间很小）。它还有一个主要特点就是它的车轮子。想想看，能利用这些轮子做什么呢？

(3) 考虑单独与复合情况的对比。当你只拥有一个物件的时候，你的使用场景可能就和你有多个物件时不一样。举个例子，一个高尔夫球可以用来打一场高尔夫球，一大堆高尔夫球则可以用来压住一块被风吹起的防水布；

(4) 考虑原版与改进版的对比。一个物件可以直接被使用，也可以改进之后再使用。如果你有能力作改进，那么它就有了更广阔的使用空间。

理论上说，还是采取“结构化创意”比较好。结构化创意能让面试官看出你有条理的思考能力，还能帮他更清晰地记住你的答案。不过，话说回来，别让你的结构化思路妨碍了你的沟通能力。

结束了头脑风暴之后，面试官可能会要求你详细谈谈其中一个想法。

1. 例题分析

针对自动贩卖机想几个创意。

这个问题可以有很多回答的方式，下面是其中一种。

> 这个问题挺有意思，请让我先考虑一会儿。
>
> 好了，首先，我们可以使用自动贩卖机来出售一些不同于普通食物和饮料的东西。基本上，我们可以出售任何小物件。我们可以把它放在酒吧卖啤酒，或是用来售卖小型的电子产品、化妆品，等等。
>
> 不仅如此。除了标准的“售卖小物件”之外，它还可以有其他用途呢。
>
> 自动贩卖机最擅长的，就是将某种东西的一件（不多不少，恰好一件）以一种全自动的方式，毫发无损地售卖出去。自动化的方式很有意思，因为这样一来，就不必有个人一直在现场服务了。还有哪儿能用上它呢？
>
> 它可以作为商店的备选。一个小型商店的店主可以在他不营业时，使用一台自动贩卖机来售卖他们的一部分商品。甚至在白天客流高峰时，用自动贩卖机来加速一些快销商品的出售过程。
>
> 或者，如果有办法可以改造自动贩卖机，让其允许人们将商品塞回去，那么它又可以用于出租商品，如手机充电器。它的美妙之处在于，如果顾客不将商品还回来，就将他们的信用卡资料记录在案。

对于那些价值较高，不能随意在货架上摆放的商品来说，这可能也有用处。很多商店会将比较昂贵的商品，如剃刀的刀片，摆放在上了锁的货架上。而顾客若需要看，他得找导购员来开锁。如果人们能刷卡直接付款，那么双方都更方便省事了。

好的，现在我再从理论上想想看。一台自动贩卖机有以下属性或者零件：

- 用以展示商品的大块玻璃窗口；
- 冷藏库；
- 自动在货架上定位到指定商品的功能；
- 信用卡处理器；
- 货币兑换机；
- 现金验钞功能；
- 防止人们把手伸入机器将商品一把抓走的功能。

另外，它还必须是一台非常沉重的自动化机器。

我们可以使用玻璃窗口来展示其他类型的商品，诸如小型的艺术作品。我可以将冷藏库像普通冰箱那样使用。

我们还可以将一些部分结合起来使用。举个例子，假如我们有一台旧的自动贩卖机，就可以把它改装成一部人们可以用信用卡提取现金的机器。

自动定位到货架上的指定行，指定列并找到商品的功能，也挺有意思的。

假定它也可以一次抓取几个商品。这样的自动化功能在一些行业里能派上大用场，尤其如果他们需要减少人工操作失误，这样的功能更是有用。大概药剂师或医生可以用上这个功能，因为他们经常要要分发药物，而且要确定拿对了药。

我想，目前这个清单差不多了。您需要我继续说下去吗？或者说，我刚才所说的这些想法中，您希望我对哪个展开详细讨论？

这位候选人选用了一种广泛的方法。他的答案首先就直接将自动贩卖机的使用情况列了出来，这些案例运用的科学技术是按照候选人期望的方式展开的。然后，他又将自动贩卖机拆分为零件，对这些零件的其他用途展开了讨论。

其实，他还可以将讨论范围拓展得更广。他原本还可以讨论多个自动贩卖机的用途，或者对改装后的自动贩卖机进行深入分析，看看它还能有什么不同的用途。

2. 样题

- 假设你是一名回形针制造商，你意识到人们不再需要回形针了。你要如何为你的回形针作市场营销？
- 尽可能多地说出酒杯的用途。
- 列出几种可将亚马逊产品或服务和汽车结合起来的方式。
- 假设有这么一个数据库，世界上任何地方只要有人说话，它就把这些语音记录为文字保存起来，而你有权限进入这个庞大的数据库。它还能极快速地响应提取信息的请求。基

于这个数据库，你能设计出什么产品或服务？

- 如何将互联网接入健身房？你能以此创建或改进什么产品或服务？
- 如果你是 Nike 的 CEO，你会想出什么新的生产线？
- 假设你在苹果公司工作，要推出一款技术含量不高的产品。你能想出什么主意？
- 突然间，有人给了你一大堆带纯平显示器的旧台式电脑，至少 1000 台，你能拿这些电脑干什么？
- 尽可能多地说出筷子的用途。
- 你会如何改善在超市购物的用户体验？假设那些听起来荒诞的技术也可以实现（假如实际上并不是不可能）。

15.5.5　定价及收益类问题

定价问题的目的是让你寻求利润最大化，利润指的是收入和成本的差值。在理想情况下，我们会用一些供求图表来说明销售量如何根据定价波动，或是对利润最高的时间段进行准确描述。

而在现实世界里，我们没有任何数据。我们的目标还是一样的，即寻求利润最大化，只是过程不同。一般人们会使用以下方法来制定商品价格。

- **成本加利润定价法**　核实你的产品成本，价格提高一点就可以了。这不太容易，一般来说都有固定成本和边际成本，所以要核计每个产品的单位成本很困难。另外，很多线上服务产品都没有直接成本，成本并不能对价格是否合理作定论。但是，你的产品成本可以提供一个最低价格（假定你想要有利润），还能帮你了解竞争对手的价格。
- **价值定价法**　对消费者来说，有些产品会有一个清晰直接的价值。在这种情况下，你可以评估一下这款产品由此能为消费者省下（或获得）多少金钱或时间。
- **竞争性定价法**　很多产品的价格是根据竞争者的价格来制定的。这种方式并不完全合理（因为消费者或许还是会选择你的竞争对手），还有一部分原因是因为懒惰（因为你不知道还能如何为产品定价）。价格定得比竞争对手低未必是好事，消费者可能会觉得你的产品质量较低，还可能挑起价格战。然而，如果你的产品定位是优质产品，而且你决定将价格制定得稍高一些，竞争性定价法仍可以作为定价的起点。
- **实验定价法**　有时候，有些公司会应用实验定价法为产品制定不同的价格，然后根据销量来调整价格。但是这种方法要小心运用，不稳定的价格可能会阻碍消费者购买，或者更严重来说，会激怒消费者。

一个考虑周详的公司可能会使用成本加利润定价法、价值定价法和竞争性定价法来定位出最佳的价位，再通过实验定价法来进行微调。

了解这些常用的定价方法后，还有一些定价模式要考虑。

- **广告支持的免费模式** 很多创业公司都会尝试这种模式，但极少成功。除非你的产品实在太出众，广告特别有效，否则光靠广告收入并不足以支持公司的日常运作。
- **免费增值模式** 在免费增值模式中，运营商会提供一个免费的基础版本，而增值版本就需要付费才能使用了。这个模式对招揽客户可能有所帮助。但是，你得时刻关注免费用户支持服务的成本和用户转化率。
- **分层定价模式** 有些公司可能会根据产品数量、客户类型或产品功能制定多个级别的价格。但你不能太过火了；对消费者来说，太多级别的价格可能太难消化了。
- **按菜单定价模式** 有些公司可能会给每个功能或服务分别定价，让消费者自己选择具体要“升级”哪些功能或服务。在这个模式下，消费者可能花费得比直接买下一整套捆绑功能套件还要多。有些消费者会喜欢这种灵活性，但也有些消费者会觉得眼花缭乱。公司要处理多个不同的功能套装，背后的支持成本也是一项挑战。
- **订阅模式** 很多服务商都会提供关于他们产品或服务的订阅服务，尤其是 Web 应用程序。有些产品同时支持购买和订阅。这就使得产品能获得那些只需临时使用，而可能不会贸然花钱买下整个产品或服务的用户。
- **免费试用模式** 短期试用可以让消费者在购买前先对产品进行体验，是一个吸引他们“上钩”的好办法。免费试用可以限制时间、用户数量或指定功能（比如说，可以导入但不可以导出）。你必须保证足够好的用户体验，让试用者喜欢上这款产品，但是又不能让体验太好，好到他们没有要升级的欲望。
- **“剃刀与刀片”模式** 有些公司会将一个物件（如剃刀）的价格压到临近或低于成本，以使一个附属物（如刀片）给公司带来额外的收入。如果消费者只能从你这儿买到这种附属物，这个模式就非常有效。如果还有其他竞争者提供可兼容的附属物，那你就要冒这样的风险：消费者从你这里买了便宜的产品， 再从竞争者那里买了附属物。谁也没得利（除了消费者）。

定价模式也可以结合起来一起使用。举个例子，一家公司可以给它的服务提供不同价位的订阅服务，根据消费者业务的规模大小来决定订阅价格，此外再按菜单定价模式提供附加功能升级服务。

1. 在线广告

很多科技公司（最出名的是谷歌和 Facebook）都拥有基于广告的商业模式，因此在面试上会经常会出现围绕广告的定价问题。

关于广告的问题看起来可能很复杂，实际上它们都能被拆解为 CTR（点击率）、CPC（每次点击费用）和用户转化率的标准比率问题。一旦你开始根据这几个比率来考虑问题，要找到答案就相当容易了。

不过，首先要对在线广告如何运作有一个基本的了解。以下是简要的介绍。

在线广告通常以两种方式来定价。

- **按点击付费模式** 只有在用户点击他们的广告时，广告主才需要付费。这就意味着如果他们的广告向用户展示了 1000 次，但只有 2 个点击，那么广告主只需要为那 2 次点击埋单。
- **按显示付费模式** 每次显示广告，广告主都需要支付费用，不管用户和广告是否有任何互动。
- **按行为付费模式** 只有当行为发生时（如有用户点击了广告并购买了产品），广告主才需要付费。这也称为一次“转化”。这种模式较为少见，部分是由于难以追踪用户的转化。

谷歌和 Facebook 都提供了两种广告模式：按点击付费模式、按显示付费模式。

此外，Facebook 也提供按行为付费的模式，允许的行为包括为页面点赞、询价和安装移动插件等等。在谷歌和 Facebook，还有其他有广告事业的公司里，一则广告的定价通常是通过拍卖来决定的。

按点击付费模式是比较常被提及的。每次点击所支付的费用就是所谓的 CPC（每点击成本）。在广告位竞拍时，基于 CPC 的报价就是你愿意支付的费用。

还有另一个关键衡量标准，就是广告被点击次数占显示次数的比例，被称为 CTR（点击率）。要计算你的广告展示 1000 次的费用（即 CPM，每千人浏览费用），可以将 CPC 和 CTR 相乘，再乘以 1000 倍。举个例子，如果 CPC 是 15 美元，CTR 是 2%那么 CPM 就是 30 美元。

一般来说，广告主愿意为一则广告（被点击一次的广告）支付的费用，最多不会超过当次点击带来的利润。要计算平均一次点击的预期利润，你得先知道点击广告的用户当中，愿意购买商品的人占这些人的比例（转化率）。然后，将转化率与每次转化的利润相乘。

```
广告最高费用
    = 平均每点击预期利润
    = 转化率 × 每转化利润
```

如果被要求计算某个指定商品的广告费用，你可能就需要对转化率和每转化利润进行估算。转化率因行业不同而大大不同，不过，对谷歌搜索式广告来说，转化率常常介于 2%与 5%之间。

2. 例题分析

如何为一款个性化的笔记本定价？

参考答案如下。

既然是个性化的，我就假设这是一款高端的笔记本，可能是真皮（或人造皮革）装订的。

首先，想想它的目标市场是什么。看起来，这款产品的市场好像就是它的消费者，但其实它还有一部分市场是将笔记本作为促销品的公司。两者都能作为面试讨论的好材料。

考虑定价问题时，有一些不同的方面需要考虑。

- ❑ 对于那些精美的 Moleskine 式样的笔记本来说，它们的成本是多少？我买过一两本这样的笔记本，它们的价格相当昂贵——大约是 15 到 20 美元。
- ❑ 其他一些个性化产品的成本是多少？我们也在这个市场里与人竞争，所以也需要知道这个。显然，个性化产品可以非常昂贵，这类产品一般很少有低于 10 美元的。我在查找个性化礼品时，价格多数是在 15 到 50 美元。

我们的顾客群一般在购买礼物上的花费是多少？送一份 20 美元的礼物在送礼行为中算是低端的，至少对那些专业买手（指的是最可能买我们这款产品的人）来说是这样。

消费者会愿意为个性化定制支付多少溢价呢？我们可以通过调查来猜测这个数据，但也并不完全精准。50%到 100%听起来就差不多可以接受了。

我们制造这款产品的成本是多少？我在这方面不是很了解。但是，根据其他产品的价格，我猜测成本应该包括笔记本本身的成本，加上个性化定制的成本。如果一本 Moleskine 笔记本在网上价格是 20 美元，那么卖给经销商的价格应该是 50%上下。因此，我们假定一本 Moleskine 笔记本的成本是 5 美元，加上个性化定制服务，最多也就多上几美元吧。

看起来，我们已经将一个精美的个性化定制笔记本的最高限价缩小到大约 35 美元了。

我猜想，这款笔记本还有一个市场，那就是一些在笔记本上刻上公司 Logo 的客户。我们可以为大量购入笔记本的客户提供批发折扣。这能够鼓励大量购买的订单，也能在许多不那么高档的促销品上给公司客户提供替代选择。

批发折扣不会降低笔记本的生产成本（因为这些笔记本已是折价生产），但可能会降低个性化印刻的成本。我们的成本大概在 7 美元左右，所以批发折扣可以降低我们的价格下限。对于大量订单，15 美元的定价看起来没问题；这给了我们 50%的边际利润率。这差不多就是我们的底线了，我们不会再往下作出太多调整了。

遵循这个定价方法，其他价格可以这样制定。

- ❑ 1～10 本：35 美元。我想，在这个范围内，还谈不上规模经济。
- ❑ 10～30 本：30 美元。
- ❑ 30～100 本：20 美元。
- ❑ 100 本以上：15 美元。

当然，我们还会根据调查数据和实际销量来调整价格。由于这是一款实体产品，我们不用对现有顾客作支持服务，所以可以定期调整价格，而不必担心引起老顾客的不满。”

这位候选人使用了多种方式，来了解合理的价格范围。她还根据产品的替代选择，对顾客支付特定费用的意愿进行了评估。在这个案例中，替代选择基本上包括其他笔记本、其他个性化定制产品和其他礼品等。她还用一些成本估算对她的定价进行验证。

3. 样题

- 说说你认为定价方式很有意思的一款产品或服务。
- 你要推出一款每播一首歌就收费一次的音乐服务。若干次播放同一首歌之后，它就默认自动购买了这首歌。你会如何为这款服务定价？
- 假设美联航决定要新推出一款“备用机会员通行证”，持有者可以无数次搭乘待飞的航班（只要机上有空位）。你会如何为这款通行证定价？
- 一家教科书出版公司决定要将他们所有的书做成电子版，并出售订阅服务。你会如何决定这款服务的价格？
- 你会如何为一款以企业为目标用户的文件备份服务定价？
- 两家商店相距 1 英里。他们销售同一款产品，但是其中一家商店的定价比另一家高出 25%。什么因素可以解释这两家商店的不同定价？
- 你会如何为一款长者专用的手机服务定价？
- 你要推出一款照片打印服务，这款服务能自动打印并向用户邮寄他喜欢的图片。你会如何为这款服务定价？
- 假如你要推出一款用于智能手机待办事项管理的软件，你会如何为这款应用定价？
- 你觉得要进行哪些分析，才能决定亚马逊是否应该，或者何时应该重新制定亚马逊金牌会员服务？

15.5.6 解决难题类问题

有些问题是完全开放的：有这么一个难题，你看着办吧。要解决这些问题，并不是从解决方案入手，而是先研究难题本身。你得找出这个难题的症结，诊断出问题的起因，然后才解决它。

某些情况下，你可能最终会走到一个岔路，面对一个不可调和的矛盾。举个例子，假设你遇到一个这样的问题：“你要推出一个照片分享的网站，而且已经试验过一个发布照片的新界面。这个新界面增加了用户停留在网站上的时长，但减少了分享照片的数量。你会怎么处理这件事？”

如果必须二选一，那就以公司的目标为转移吧。每个公司不止目标不同，就连一个产品在不同时期也可能有不同的目标。举个例子，一个新创业公司最初可能会将获取用户作为首要目标，但后来就会优先考虑收入。

根据面试官的考验目的，她可能将这个问题作为一次你来我往的练习：你来问问题，搞清楚难题所在，她则提供答案。或者她可能想要考验你的直觉反应，看你在真实世界里怎么解决这个

难题。

将难题拆解开来，可以帮助你找到问题的真正原因。 以下是一些常见的问题和引起它们的原因。

- **利润下滑**　起因于收入下滑或成本上涨。
- **收入下滑**　起因于销量下滑或价格下降。这可能意味着用户在不同产品级别上的购买行为的转变。
- **销量下滑**　起因于新客户量的减少或现有客户粘性的下降。
- **新客户量减少**　起因于客户流量减少或转化率下滑。二者可能分别源于网站的回头客量和新访客量的变化。
- **成本上涨**　可起因于固定成本或附加成本的上涨，而这二者可以是起因于供应商价格上涨、经销商改变利润结构、退货量增加或各种各样其他方面的变化。
- **流量减少**　可起因于新访客量减少、回头访客量减少或这二类客户停留时间的减少。
- **新访客量减少**　可起因于搜索流量减少、推荐流量减少或直接访问量减少。

“你会如何搞清楚利润下降的原因？” 一个这样的问题可以将利润拆解为收入、销量、新客户量、用户转化率，等等。关键是要确定问题的根源。

一旦搞清楚了是哪个变量在发生变化，就需要诊断问题的起因。可能是出于任意原因，这里有一些建议，你可以由此下手：

- 所有区域都发生了同样的变化吗？
- 我们有多少条产品线？这个变化是否在所有产品上都发生了？
- 据我们所知，同行竞争者是否遭遇了同样的问题？
- 其他相关产品是否遭遇了同样的影响？
- 是否与季节性有关？
- 我们是否对生产线做了什么变更？
- 是否有新竞争者进入市场？
- 如果将客户按新客户和回头客分开，我们会看到什么不同？
- 用户粘性的数据怎么样？
- 我们的客户最近有什么反映？最近我们收到的投诉是否比平时多？
- 是否有留意到推荐流量的任何变化？

诊断出了这个问题，面试官可能会要求你解决这个问题。在着手开始解决之前，要搞清楚你的解题目标。

1. 例题分析

假设你在亚马逊的服饰部门工作，你发现牛仔裤的销量在过去三个月中稳步下滑。你如何弄

清楚其背后的原因?

你可以这样回答这个问题：

这个问题挺有意思。好嘞，牛仔裤的销量下滑了，假定这里指的是大幅度销量下滑。我需要知道更多信息，才能找到原因。

三个月？这事儿有点蹊跷。一定是在某些方面发生了一些事情，这些方面可能是牛仔裤、服饰、亚马逊、电子商务，或者只是跟具体的这段时间有关系。

我认为，可以排除电子商务或 Amazon.com。因为，如果这两者受到了重创，我们肯定会知道，而且那个时候我们要担心的就不只是牛仔裤销量下滑这个问题了。同样，我们也排除掉亚马逊服饰销量的下滑，因为，假如是整个服饰类别的问题， 那我们就不会特别来担心牛仔裤的销量了。

时间问题也挺有意思的。将销量与和三个月前作对比，并不是个好主意。一年下来，牛仔裤的销售应该都不是稳定的。即使确实要作对比，也该和上一年的同一时期对比，以此排除季节性的原因。但让我们假设，这里面还是存在一些问题。

你也说过，这个销量是稳步地下滑。这就意味着，问题应该不会是由于某个 UI 的变动之类的原因。因为在那种情况下，销量会突然下滑，而不是稳步下滑。

这个时候，我想要搞清楚牛仔裤类别内部的销量。

- 是否所有牛仔裤的销量都下滑？还是只有某些特定的品牌?
- Zappos 上的牛仔裤是否遭遇了一样的情况？由于 Zappos 属于亚马逊旗下，我们应该能获取这个信息。

因此我们不妨假设这个问题指的是亚马逊上所有牛仔裤，而 Zappos 并不受影响吧。这样，我们就能把问题归结到先前说的一些特定的原因上了。

销量是一个关于访客和转化率的函数。如果销量下滑了，就说明与其中一个或两个变量有变化。

我们可以将访客与转化率按访客类型进行划分：亚马逊搜索流量（在 Amazon.com 上进行搜索的用户）、浏览流量（在 Amazon.com 上浏览并找到目标商品类别的用户）、外部搜索流量（搜索引擎流量，如谷歌）和直接流量（直接打开特定产品页面的用户）。

我们的目标是，在这些访客类型中，锁定那些在访客量或用户转换率上有所下滑的类型。举个例子，如果外部搜索流量下滑了，那我们可以对可能造成 PR 值下降的改动进行调查。

如果在这些类型的用户量中看不到异常，就得从别处寻找可能影响销量的东西。举个例子，这可能是因为牛仔裤的价格大幅上升。还有可能（我倒希望不是）是因为系统的搜索功能或购买流程出了错，影响了销量。

这就是我来处理这个问题的基本流程。不知道您想要我深入分析哪一方面?

回顾刚才这个例子，这位候选人牢牢抓住了这个问题的措辞：这是“某产品类型”销量的“稳步”下滑。要寻找可能造成问题的原因，这是一个很好的线索。

同时，这位候选人还在“概述一个通用的方法来解决问题”和“使用适当的直觉来引导解题方向”（如牛仔裤一般不会突然间就卖得不好）之间保持了平衡。

2. 样题

- 你注意到你网站上的广告收入出现了大幅下降。你将如何着手去搞清楚发生了什么事情？
- 你将本月与上月的用户流量进行对比，发现本月流量比上月下降了 10%。你会怎么做？
- 一家杂志公司向你寻求帮助。他们明白出版是一个麻烦的行业，但是他们的销量下降了 10%，而他们最直接的竞争对手只下降了 5%。你会如何处理这个问题？
- 一个 Facebook 的某页面有 10%的机会报错。这是怎么回事？
- 你的 VP 量太低，你得在 4 年内将收入翻一番。你会如何制定计划以达到这一目标？
- 你拥有一个专营体育用品的小型电子商务网站。去年你净赚了 20 万美元，而今年的收益只有 8 万。到底是怎么回事？你会如何找出原因？
- 你网站上的产品价格有三种级别：免费、标准、高级。如果你发现高级产品的销量下降了，而标准产品销量上升了，你会怎么做？
- 两个孩子分别经营一个柠檬水站，他们之间相距几个街区。有什么因素可以使得其中一个孩子经营得比另一个孩子好得多呢？
- 你将要针对你公司的网站推出一项重要的 UI 改动。你会对哪些衡量标准进行监控，以在有问题时能及时知悉？
- 你注意到，某关键词在西班牙的谷歌关键词广告收入连续下降了 30 天。这个关键词非常重要，所以谷歌要求你搞清楚原因。你会如何解决这个问题？

第16章 编码问题

还想着面试时不用写代码就能过关呢？不要太早下结论。

很多公司，包括谷歌、亚马逊和微软，有时候都会向产品经理应聘者提出编码和算法方面的问题。这些问题范围广泛，既有简单直接的编码问题，也有比较复杂的算法问题（可能会要求你写代码，也可能不会）。

不过他们对产品经理应聘者的期望通常要比开发人员岗位的期望低。很多面试官能接受伪代码，但也有不接受的。

16.1 谁需要写代码

一般来说，你写代码的时间越近，越有可能被问到编码方面的问题。大多数公司一般会在面试计算机科学专业毕业生和开发人员时提出编码的要求。

之前从来没写过代码？那你可能不会被问到编码方面的问题，但仍然需要掌握一种解决问题的办法。为了以防万一，还是阅读下这一章的内容。其中有些术语可能你不懂，但知道如何解决这些问题还是有好处的。

16.2 你需要知道什么

如果你刚修读过一门很专业的计算机科学课程，可能你已经掌握大部分编码相关的知识点了。除非你发现自己还有很大的差距，否则就可以专心准备面试问题，不用再重新学习这些知识点了。

这里有个你应该掌握的知识点的快速清单。

注意：我们直接跳到了大O上来解释运行时长。如果你不知道什么是大O，或者对它的含义感觉很模糊，可能要再温习一下这个知识点。

16.2.1　数据结构

顾名思义，数据结构是一种保存数据的结构。根据目标不同，有很多种不同的数据保存和组织方式。

如果按照面试中重要程度大致顺序的降序排列，常见的数据结构如下。

1. 数组

数组是保存一组对象最简单直接的方式，它把数据项存在一个简单的对象列表中。如果你知道索引，查找相应对象的速度很快，不知道索引时则很慢。比如说，获取列表中第 12 个人的信息很快捷，但找到所有名为 Alex 的人则很慢（因为你要查看所有人的名字）。

在大多数编程语言中，数组在创建后长度不能“增长”。必须提前指定数组的长度，且之后不能再改。

优秀练习题：

16.1　给定一个已排好顺序的正整值数组，末尾有一个空位（0），按顺序插入一个元素。

228 页

16.2　逆序排列数组中的元素（不创建新数组）。

230 页

2. 散列表

用散列表（有时也被称为“字典”或“散列图”）可以将一个“键”映射到一个“值”上。键通常是数字或字符串，而值可以是任何类型的对象。

这是一个非常实用的数据结构，因为它的查找速度非常快。对于面试而言，我们通常会假定散列表插入和查询的时长是 $O(1)$（不管数据量有多大，时长都是恒定的），虽然实际上并不完全是这样。糟糕的散列表查询时长可能会变成 $O(N)$。

可以利用它将一个人的 ID 值映射到包含其他相关信息的某个对象上。

优秀练习题：

16.3　假定有两个字符串链表（A 和 B），其中包含的字符串都是唯一的，写一个程序确定 A 是否为 B 的子集，也就是检查 A 的所有元素是否都在 B 中。

230 页

16.4 给出一个含有销售数据的二维数组，第一列是产品 ID，第二列是数量。写一个函数，以这个数据列表为输入，返回一个新的二维数组，其中是每个产品 ID 及其总销量。

示例：

```
输入：
211,4
262,3
211,5
216,6

输出：
211,9
262,3
216,6
```

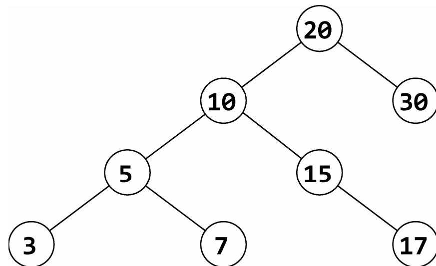
232 页

3. 树和图

图是由边连接的一组节点。并不是所有节点都要连接在一起，你可以有两个完全独立的子图，并且边也可以是“有方向”或“无方向”的。有向边可以被看做单行道，而没有方向的边就像双向道。如果图是有方向的，那从 v 到 w 的边就不能等同于从 w 到 v 的边。同理，可以从节点 n“走到”节点 m，但反方向不行。

树是图的一种，任何两个节点之间都有且只有一条路径。因为任意两个节点之间只能有一条路径，所以树中不可能有环路。

树有很多种形式，但目前最常见的是**二叉树**。二叉树中的所有节点最多只能有两个子节点。我们将这些节点称为左节点和右节点。跟所有树一样，二叉树中也不能有任何“环路”（没有能回到节点自身的路径）。因为这些限制，一个二叉树可以表示为下面这种严格的等级形式：



一般我们处理的是**二叉查找树**。在二叉查找树中，一个节点左侧子树中所有节点的值都比该节点的值小，而右侧子树中节点的值都大。以上就是一个二叉查找树的例子。

如果一个二叉查找树是平衡的（一般我们处理的都是平衡的二叉查找树），插入一个元素，

以及查找元素的时长都是 $O(\log n)$，其中 n 是节点数。

优秀练习题：

16.5 往二叉查找树里插入一个元素（按顺序），假定这个二叉查找树里包含的是整数。

233 页

16.6 给定一个包含整数的二叉查找树，计算所有数的和值。

233 页

4. 链表

和二叉树一样，链表也是由节点组成的数据结构，每个节点都有一个指向其他节点的指针。在单向链表中，每个节点都只有一个指向下一个节点的指针。在双向链表中，节点有一个指向前一个节点的指针，还有一个指向下一个节点的指针。如果在链表中存在回路，通常会认为是一个严重的问题，并且可能会破坏链表的结构。

向链表前面插入一个节点的时长是 $O(1)$。然而，如果链表是有序的，在你希望按顺序插入节点时，时长是 $O(N)$，其中 N 是节点数。这是因为你必须先找到正确的位置，而这要求你遍历整个链表。

不管链表是否已经排好序，在其中查找节点的时长都是 $O(N)$。

优秀练习题：

16.7 在已排序链表中插入一个节点（按顺序）。（别忘了新元素出现在链表头或尾部时会发生什么！）

235 页

16.8 对一个只包含 0 和 1 的链表排序，也就是将链表中的所有 0 都放到 1 之前。

236 页

5. 栈

栈这种数据结构精确定义了数据插入和移除的顺序。在添加元素，或“压入”时，是在栈顶插入。移除元素是从栈顶弹出。

栈是一种 LIFO（后进先出）的数据结构，因为最后（最近）添加的元素是第一个被移除的。

就像我们现实生活中遇到的层叠板一样。当你向层叠板上放板子时，是放在最上面的；而当你拿掉一块板子时，也总是从上面拿。

栈的插入和移除操作时长都是 $O(1)$。一般不会在栈中找有特定值的元素，因为那样要一个接一个地移除所有元素。如果要做这样的操作，最好别用这种数据结构。

优秀练习题：

16.9 写一个函数，以栈为输入，然后返回一个元素顺序相反的栈。

239 页

16.10 写一个函数，从栈中移除所有偶数，返回到原来那个栈，而不是新建一个。

240 页

6. 队列

队列跟栈相反，它不是用 LIFO（后进先出）原则移除最新的数据项，而是移除最老的数据项。是“FIFO”（先进先出），因为第一个添加的数据项会被先移除出去。

这就像现实生活中排队（或流水线）一样。当人们排队买电影票时，第一个排上的人是最先得到服务的，所以这个数据结构就叫队列。

从队列中插入（或“入队”）和移除（或“出队”）数据项的时长都是 $O(1)$。跟栈一样，一般不会在队列中查找某项元素。

优秀练习题：

16.11 写一个函数检查两个队列是否相同（以相同顺序排列的相同值），修改/销毁这两个队列也没关系。

241 页

16.12 写一个函数，从队列中移除第 13 个元素（但要保证其他元素还按原来的顺序排列）。

242 页

16.2.2 算法

如果你有计算机科学学位，应该知道仅是高级算法就可以写个上百页。但我们不会那么干，因为面试中几乎不会问到那些算法。即便应聘开发人员也不太可能会被问到，比如 Dijkstra 算法，因为面试官更关心你能不能创建出新的算法，而不是能否记住已有的算法。

然而，有几个基本算法对开发人员来说是必须掌握的，甚至对产品经理来说也是如此。它们出现的频率很高，所以应该花些时间记住。

1. 排序

数组排序方法中最常见的两种分别是快速排序和归并排序。其他排序算法，冒泡排序、插入排序、基数排序等一般效率稍低，或者只在特定条件时使用。

- **归并排序**分别对数组的左半边和右半边排序，然后合并数组。通过归并排序算法（递归）。拿到左半边，将它分成两半，对两部分分别排序，然后再归并到一起。再在右半边做相同的操作。归并排序的一般情况和最坏情况下的时长是 $O(n \log(n))$。
- **快速排序**会随机选择一个“轴”元素，然后根据数组中元素相对轴元素的大小重新排列。接下来再处理轴左边（全都小于等于轴元素）和轴右边（全都大于轴元素）。两边应用的策略都一样：选一个轴元素，重新排列，然后在每边再选新的轴元素。快速排序一般情况下的平均时长是 $O(n \log(n))$，但最坏情况下的时长是 $O(n^2)$。如果一直选到坏“轴”（非常小或非常大的元素）就会出现最不理想的情况。

这两个算法都采用了“分成两部分然后重新应用算法”的方式。

其他排序算法都比较初级，就好像是你要对一沓纸排序。

- **插入排序**在数组头部维护一个排好序的元素子表（最初是 0），然后看未排序子表的头部。如果这个元素比已排序子表的最后一个元素大，就把它留在那里，然后增加已排序部分（因为这个元素已经是按正确的顺排列了）。相反，如果这个值小的话，则把它挪到已排序子表中。未排序部分每次都会缩小。这个算法会对未排序部分的每个元素重复这一步骤，直到整个数组完全排好序。插入排序在最好情况下的时长是 $O(N)$（如果数组已经排好序了），但在预期和最坏情况下是 $O(N^2)$。
- **冒泡排序**是相当简单直接的算法。它会对列表重复迭代，交换每对顺序不对的元素的位置。当一次完全的迭代没有出现任何交换时，数组就排好序了。最好情况下的时长是 $O(N)$（如果数组已经排好序了），在预期及最坏情况下的时长是 $O(N^2)$。

面试过程中一般不会让你实现一个算法，但知道它如何对数据排序还是有用的。面试官有时会要求产品经理对数据排序。那些最近拿了计算机科学学位的人很可能要实现一个性能更优的算法，而其他没有计算机科学学位的应聘者可以凭比较初级的方式过关。

优秀练习题：

16.13 给定两个已排好序的数组，写一个函数，按顺序将其中的元素合并到一个新数组中。

242 页

16.14 实现插入排序。

244 页

2. 二分查找

二分查找是从一个已排序列表（通常是数组）中查找一个值的算法。在二分查找时，我们用所要找的值跟列表中间点上的值进行比较。因为列表是有序的，所以我们能确定该值应该在被比较元素的左侧还是右侧。然后在左侧或右侧查找，不断重复这一操作：比较子列表的中间点，在子列表的左侧或右侧查找。

因为不断将数据集分成两半，这个算法在一般情况和最坏情况下的时长是 $O(\log n)$。

其实我们在现实生活中经常使用二分查找法，只是没意识到。比如你有一沓按名字排序的学生试卷。如果有一个名为 Peter 的学生，你会从一沓试卷的最上面开始找吗？应该不会，你应该是直接翻到中间，然后比较。如果看到 Mary，你知道还要继续，然后你可能会将那一半试卷再分一半继续查找。

二分查找是一种很流行的算法，因此也是一个需要掌握的重要概念。很多算法都是基于二分查找法的。

优秀练习题：

16.15 实现二分查找。即给定一个已排序的整形数组和一个值，找到那个值的位置。

244 页

16.16 给出一个已排序的整型数组，然后旋转一下，其中的元素都是唯一的，找到其中的最小值。比如说，数组可能是 `6，8，9，11，15，20，3，4，5`。很明显最小值是 `3`。

245 页

3. 图搜索

图的搜索有两种常用算法：深度优先搜索和广度优先搜索。

在深度优先搜索中，会先完成节点的第一个子节点上的搜索，然后再去搜索第二个子节点、第三个子节点等等。比如说，一个有两个子节点 `A` 和 `B` 的节点，如果我们要搜索值 `v`，会先完成 `A` 上（以及连接到 `A` 上的节点）的搜索，然后才会到 `B` 上搜索。因为我们在广度上扩大范围之前会不断深入，所以它被称为“深度优先搜索”。

在广度优先搜索中，我们先在广度上搜索，然后再深度搜索。如果从初始节点 `R` 开始，我们首先会检查 `R` 和与其直接连接的所有节点（这些节点称为“子节点”）。然后再扩展我们的搜索范围，搜索所有与 `R` 的子节点连接的节点。一直重复这个过程，直到找到目标值，或者完成整张（子）图的搜索。

在这两种算法中，一定要小心千万不要进入环路。因此，如果图中有环路，也就是说，如果

从一个节点到另一个节点的路径不止一条，那我们就要将节点标记为“已访问”，从而确保我们不会重复搜索相同的节点。在树中不会出现这种问题，因为树中没有环路。

注意，一个图可以有两个完全分开互不相连的部分。如果遇到这种情况，我们在查找数据项时需要在所有部分上执行搜索算法。

优秀练习题：

16.17 使用深度优先搜索，检查一棵树中是否有某个值。

247 页

16.18 编写在二叉树上进行广度优先搜索的伪码（尽可能详细）。

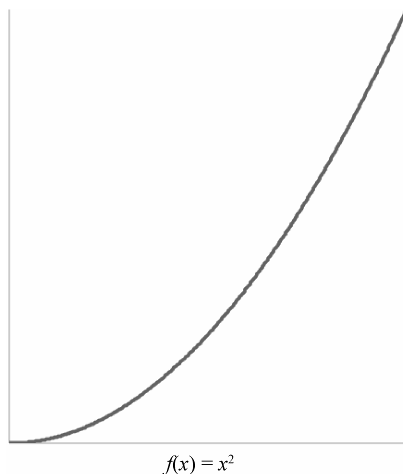
247 页

16.2.3 概念

1. 大 O 表示法

大 O 表示法是用来表示算法效率的。如果你要从事编码方面工作，那一定要理解大 O 表示法。毫不夸张地说，它是我们用来表示效率的语言。

使用大 O 可以表示不同算法之间的差异。比如说，如果你正在做一个社交网站，想显示两个人有多少个共同的好友，你可能会逐一查看我的朋友列表中的每一个人，看他们是否也在你的朋友列表中。这个操作所用的时长可能是 $O(N^2)$，其中 N 是用户朋友的平均数。那就是说，如果你要计算这种方式在朋友数量渐渐变大时所用的时长，将会看到一个形如 $f(x)=x^2$ 的函数图形。这个成本非常高，你需要拿出一个更好的实现方法。



$f(x) = x^2$

用大 O 可以清晰简洁地传达这种信息。它能表示一个程序在输入数据规模发生变化时执行时长的变化情况。即随着输入变得越来越大，程序要多花多长时间？只是多花一点儿时间吗？会长很多吗？时间会随着输入的大小呈指数级增长吗？（天哪！）

假设有一个函数 `foo`，处理一个大小为 N 的数组。如果 `foo` 的执行时长是 $O(N)$，那么随着数组大小的增长（即随着 N 的增长），`foo` 所用秒数也是呈线性增长的。

信鸽与互联网

这是一个真实的故事。

2009 年，南非一家名为 Unlimited 的公司实在无法忍受其 ISP 缓慢的网速而制造了一桩新闻，用一种滑稽的方式展示了其网速的糟糕程度。他们发起了一场信鸽跟 ISP 之间的“竞赛”，把一个 U 盘绑到信鸽的脚上，然后让它飞到 50 公里之外的办公室。与此同时，公司会通过网络向同一个办公室传送相同的数据，戏剧化的是信鸽竟然赢了，差距还很大。

这个 ISP 很可笑，是不是？一只鸟传输数据的速度都比他们快。一只鸟！

他们的网速可能确实很慢，但也可能并不慢，这个实验实际上证明不了什么。不管网速有多快，你都可以选一个数据量让网络或者信鸽赢。

原因很简单。

信鸽在腿上带着一个 10GB 的 U 盘飞 50 公里需要多长时间？假定要 3 个小时。

那么在网上传输 10GB 的数据需要多长时间？假设你的网速相当快，传 10GB 只要 30 分钟。好，那么传 100GB 的数据时，所需时间要超过 3 个小时。

让同一只信鸽“传输”100GB 数据需要多长时间？还是 3 小时。信鸽的传输速度跟数据量没关系。U 盘很轻，并且能装下很多数据。（当然，这里对问题作了简化。当数据足够多时需要很多 U 盘，因此也需要很多信鸽。）

所以，就是这样，信鸽打败了网络！

信鸽的传输时间是常量。网络的传输时间和数据量成正比：两倍的数据就需要两倍的时间。

用大 O 表示这种时长，我们会说信鸽的时长是 $O(1)$，它传输 N 千兆字节时的时间变化是 1，也就是说它根本不变。

网络的传输速度是 $O(N)$，意味着它的时间变化跟 N 成正比。

大 O 提供了一个描述过程执行时长相对其输入数据如何变化的方程式。它描述了变化趋势，它不是定义准确的执行时长，因为大 O 值比较大的过程在特定的输入上可能会更快。

现实生活中的大 O

现实生活中很多“操作”的时长都是 $O(N)$。比如开车就可以看做是 $O(N)$的，随着距离 N 的增加，开车的时间也呈线性增长。

什么事可能不是 $O(N)$的呢？

假定我们要邀请很多人（包括你）参加一次晚宴。如果我们邀请的人数增加了一倍，那你握手的次数就要增加一倍。你跟每个人握手的总时长可以表示为 $O(N)$。如果我把客人的数量翻一番，你就要花两倍的时间，这是线性的，或者说 $O(N)$的增长。

现在我们假定每个人都要握手，但由于某种奇怪的原因，一次只能有两个人互相握手。随着 N 的增加，这种打招呼握手的时间要增加多少？好的，你的工作用掉 $O(N)$的时长，但其他所有人也都是。这个时长就要呈 $O(N^2)$的指数级增长，因为正好有 N^2 对。

丢弃常量

如果你足够留意，可能会说“等一下！不是 N^2 对。人们不会跟自己握手，并且你把每一对都数了两次。实际上是 $N(N-1)/2$ 对。所以应该是 $O(N(N-1)/2)$。”

你绝对是正确的，确实是 $N(N-1)/2$ 对（即 $0.5 \times N^2 - 0.5N$），但我们仍说它是 $O(N^2)$的。

大 O 的波动性和模糊性非常大，我们要表示时间的大致变化趋势，不是要精确计算某件事情所用的秒数。

我们会丢掉常量因子，所以 $O(2N)$跟 $O(N)$一样。还会丢掉常量的加或减，所以 $O(N-5)$也变成了 $O(N)$。综合来讲，就是说 $O(N^2+N)$应该写成 $O(N^2)$。想一想，如果 $O(N^2)$跟 $O(N^2+N^2)$是一样的，那么这两个值之间的 $O(N^2+N)$也应当被看做相同的值。

这是一定要理解的知识点。你决不能说一个算法的时长是“$O(2N)$”，这不是比 $O(N)$“更精确”或“更好”的答案；它只是徒增困扰。时长是“$O(2N)$”的算法就是 $O(N)$，就应该这么表示。

下面哪个表达式跟 $O(N^3)$相等？

$O(3N^3)$
$O(N(N^2+3))$
$O(N^3-2)$ $O(N^3+N \lg N)$
$O(N^3-N^2+N)$
$O((N^2+3)(N+1))$

这些全是！

把常量丢掉，只保留最重要的部分。

16

多个变量

还是以握手为例。假定我们的晚宴邀请了男士和女士。所有男士彼此都认识，所有女士彼此也都认识。因此人们只会跟异性握手。

假定我们还是在一个奇怪的地方，一次只能有一对握手，你怎么表示这要花多长时间？

不要说是 $O(N^2)$。假如有 100 位男士和 1 位女士，增加一位男士只会增加一次握手次数，但增加一位女士将会增加 100 次握手次数。因此，所用时间实际上并不跟人数的平方成正比。

这里有不同的“变量”，跟我们增加哪个是有关系的。这种情况的正确表达方式是用两个变量。如果有 M 位男士和 W 位女士，这样我们相互介绍所用的时间是 $O(M \times W)$。

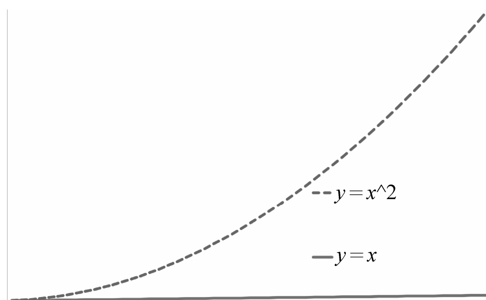
如果女士互相之间都认识，但男士谁都不认识会怎么样？这样我们会说相互介绍的时间是 $O(M^2+M \times W)$。注意，我们并没有丢弃额外的 $M \times W$；它是不一样的变量，并且它跟时间有关系。

这个为什么有关系（以及它为什么没有）

假设我们有两个处理某些数据的函数。函数 `foo` 的时长是 $O(N)$，函数 `bar` 是 $O(N^2)$。在一个给定的数据集上（比如说一个人的专业技能列表），哪一个会更快？

实际上我们无从得知。

`foo` 的运行时间增长跟 $O(N)$成正比，而 `bar` 的增长跟 $O(N^2)$成正比。所以 $O(N^2)$这条线最终会超过 $O(N)$。



然而我们不能判断在特定数据集上的情况。$O(N^2)$在比较小的数据集上可能更快；它可能还没超过 $O(N)$那条线。此外，即便在非常大的数据集上，在“超过”发生之后，也有可能会出现例外。可能在 N 能被 1000 整除时，`bar` 的代码会碰到一种特殊的情况，突然变得非常快，这些我们都无从得知。

不过大 O 仍然有它的价值，只是在使用时要非常小心。

大 O 是让我们说：“一般而言，随着数据集的尺寸变大，这个算法要比其他方式快得多。”

我们还可以说："你要运行这个 $O(N^2)$的算法，N是我们网络中文件的数量？对不起，那个不行。"那要重要得多。

此外，它还是一种不依赖于系统架构或所用技术的效率表述语言。没有大 O，我们很可能只能按秒来讨论效率，当你用的是不同系统时，那种讨论没什么意义。

对数和大 O

可能你已经注意到了，在讨论问题时，我们（和其他人）会提到 $O(\log(N))$ 或 $O(\lg(N))$，但没有特别指明是 $\log_2(N)$还是 $\log_{10}(N)$。因为这没关系，这两种对数之间的差异是常量因子：$\log_b(N)$ 等于 $\log_k(N)/\log_k(b)$。因为大 O 不关心常量因子，所以我们也不需要关心对数的底数是什么。

大 O 的空间复杂度及更多用法

大 O 的概念不仅可以用来评估运行时长，实际上它还可以描述一个算法要用多少内存。

比如说，假定有个算法要创建并初始化一个 $N \times N$ 的矩阵：

```
1  int[][] a = new int[N][N];/*NxN 矩阵*/
2  for i from 0 to N {
3     for j from 0 to N {
4        a[i][j] = i + j
5     }
6  }
```

这个算法的时间复杂度和空间复杂度都是 $O(N^2)$。

注意：如果你上过算法课，应该还记得，从技术角度讲，大 O 表示的是上边界。所有复杂度为 $O(N)$的算法都可以说是 $O(N\text{^}2)$的。要描述确切的运行时间，我们应该用大 Θ。

按照大 O 的官方数学定义，确实是这样的。然而在算法课之外，这种区别已经被忘记了。

问题示例

现在我们来看一些例子（用伪码编写）。你能找出每个问题的时长吗？

例 1

请看下面这段从 0 输出到 n 的代码。

```
1  for i from 0 to n {
2     print i
3  }
```

这个时长应该是 $O(n)$。也就是说，如果我们输入很多不同的 n 运行这段代码时，运行时间应该跟 n 成正比。

例 2

这段代码呢？

```
1   sum=0
2   for i from 0 to n {
3      sum = sum + i
4      for j from 0 to n {
5         sum = sum + j
6      }
7   }
```

这段代码的时长是 $O(N^2)$。这段代码有两个 `for` 循环，每个都是从 `0` 到 n。第 5 行执行了多少次？$O(N^2)$。这段代码的运行时间会以 $O(N^2)$的比例增长。

例 3

下面这段代码用了两个变量。它的运行时间是什么样的？

```
1   /*假定 A 和 B 都是数组*/
2   for i from 0 to A.length {
3      int j = 0;
4      while ( a[i] != b[j]) {
5         print a[i]
6         j = j + 1
7      }
8   }
```

这个是 $O(a \times b)$，其中 a 是 A 的长度，b 是 B 的长度。尽管内部的 `while` 循环可能会提前终止（找到 `a[i]`），但我们预期的情况是它会遍历 B 中的所有数据项。

例 4

还有个更难的例子。

```
1   int i = N;
2   while i >= 1 {
3      print i
4      i = i / 2
5   }
```

想一想这个 `while` 循环会做什么。这个循环会做一些事情（输出一个值），然后一直除以 2 直到它小于 1。

我们要把 N 除多少次 2 才能让它小于 1，从而让 `while` 循环终止？如果从相反的方向来说，我们可以说：我们要把 1 乘以多少次 2 才能得到 N？如果这个值是 x，那么 $2^x=n$，因此这个 `while`

循环会重复 x 次。

现在我们只需求出 x：

```
2^x = n
log(2^x) = log(n)
x log(2) = log(n)
x = log(n) / log(2)
```

所以这段代码的运行时间是 $O(\log(n))$。

应该注意：如果有东西一直分成两半，它的时间复杂度是 $O(\log(N))$。

2. 递归

如果一个函数可以调用其他函数，然后再调用它自己，这就是递归。

递归这种策略可以用来解决大量问题。如果一个问题的解法能被定义成子问题的解法，就可以用递归解决。

比如斐波那契数列问题。什么是 $n!$（n 的斐波那契数列）？ $n!$是 $n \times (n-1) \times (n-2) \times \cdots \times 1$。可以说 $n!$就是 $n \times (n-1)!$。

因此可以用极少的代码计算 $n!$。

```
1   int factorial(int n) {
2      if (n == 0 or n == 1) { /*基本情况*/
3         return 1;
4      } else {
5         return n * factorial(n-1);
6      }
7   }
```

基本情况（或终止条件）极其重要，没有它这个函数就会一直运行下去。

这里还有一个递归函数的例子：计算第 n 个斐波那契数。你可能还记得，第 n 个斐波那契数，$f(n)$就是 $f(n-1) + f(n-2)$。

```
1   int fibonacci(int n) {
2      if (n == 0) {
3         return 0;
4      } else if (n == 1) {
5         return 1;
6      } else {
7         return fibonacci(n-1) + fibonacci(n-2);
8      }
9   }
```

这是个天生的递归函数，因为第 n 个斐波那契数肯定是由比其更小的子问题定义的。

内存占用

所有能用递归解决的问题也能用循环解决（非递归的），尽管有时候用循环实现起来更加复杂。然而递归有个缺陷，即它的内存占用。

再看一下这个例子：

```
1   int factorial(int n) {
2      if (n == 1) { /*基本情况*/
3         return 1;
4      } else {
5         return n * factorial(n-1);
6      }
7   }
```

它的时长是 $O(N)$，并且任何解法都是。那内存占用情况呢？它的内存占用情况也是 $O(N)$（假设编译器没做过优化）。

方法 `factorial(n)`调用 `factorial(n-1)`，然后它又调用 `factorial(n-2)`，以此类推。注意，在 `factorial(n-1)`完成之前，`factorial(n)`是无法完成的，同样，在 `factorial(n-2)`完成之前，`factorial(n-1)`也是无法完成的。

因此，在某个时点，我们的“调用栈”中有 `n` 个函数。

```
factorial(0)
factorial(1)
...
factorial(n-1)
factorial(n)
```

其中每一个都会占用一些内存。因此，在某个时点会占用 `n` 块内存。这就是说当用递归实现这个程序时，时间复杂度和空间复杂度都是 $O(N)$。

这就是递归的缺陷：递归调用比较占内存。

16.3 如何评估

有些编码和算法方面的问题可能会相当简单，但也有相当数量的问题实际上很有难度。它们当然应该有难度：面试官没办法用简单的问题筛选出卓越的应聘者。

实际上你并不需要把那么困难的问题马上解决掉，但很多应聘者还没意识到这一点。如果你能做到，很好，但通常那是不现实的。解决一个困难问题经常要耗去你整次面试的时间，甚至有时还需要面试官指点。

对你表现的评价不是你是否答对了问题，这意义不大。相反，对你的评价更多是定性的，坦

率地说，是主观的。

- 你解决问题的意愿有多强？如果你被吓到并放弃了，那就很危险。面对有难度的问题时感到兴奋的应聘者才是面试官想要的，这样的人一般会成为好员工。
- 你解决问题的速度有多快？
- 你的算法有多优异？
- 你是如何解决问题的？
- 你需要多少帮助？
- 你的代码有多清晰？
- 你讨论问题时在沟通上的表现如何？你对面试官的反馈和指导是如何响应的？

所有这些都不是非黑即白的决定性因素。

没人给面试官一个多快是“快”的指标，或者出多少 bug 是“爱出问题的”，那么她是如何确定你表现如何的呢？

她要间接地比较你和其他应聘者的表现，然后才做决定。也就是说，在她第一次问问题时，她也不太拿得准 X 分钟是快还是慢。随着问的人越来越多，她就越来越清楚。如果其他人一般要花 20 ~ 30 分钟来解决一个问题，而你能在 10 分钟内完成时，她就知道你很快。

因为评价是相对的，所以你也很难判断自己在面试中表现如何。你可能觉得自己倍受煎熬，但你不知道其他应聘者可能更煎熬。

16.4 如何处理

编码和算法问题是用来测试你解决问题的技能的，因此你要向面试官展示你是如何处理问题的。

下面这些处理方式很有效。

(1) *澄清问题*：确保你理解了问题问的是什么。提问题验证所有的假设，比如说，如果是二叉树，那么是二叉搜索树吗？树是平衡的吗？你甚至应该用自己的语言把问题重复一遍。

(2) *到白板上去*：当你听到一个问题时，到白板那里根据这个问题创建一个例子。你的例子应该具有充分的一般性，对你解决问题有帮助，并且应该避免特殊情况。

(3) *说出来*：要跟面试官说出你的解法并做头脑风暴。如果你想用蛮力解决问题，但觉得它不够好，那就勇敢地跟面试官解释那种解法，最起码你会获得一个解决问题的起点。

(4) *批判性地思考*：拿出一个算法后，要全面地考虑它是否真的有效。时间复杂度是多少？你还能做得更好吗？它真的能解决问题吗？有任何会导致它失败的情况吗？

(5) *有条不紊地编码*：只要你和你的面试官觉得可以进入编码环节了，就可以走到白板前开

始编码。如果有必要，你可以先写一些伪码。编码时，一定要确保你确实了解自己在做什么。如果你在编码时感到疑惑，后退一步想想你的算法，欲速则不达。

(6) 测试和修订：不能因为没有电脑就不做测试，你必须测试一下代码。在这种情况下，你应该用边界情况和正常情况过一遍代码。当你找到 bug 时，你会找到的（面试过程中几乎不会出现没有 bug 的代码），想想是什么引发了 bug，然后小心地修复。

注意，编码是第 5 步，不是第一步。不要一听到问题就走到白板前开始写代码。花时间跟你的面试官头脑风暴一个解决方案。算法部分通常要占用面试的大部分时间，有时甚至是整个面试。

记住：如果你在解决问题时感觉很挣扎，那是正常的，这些问题本来就是要有难度的。

16.5 开发一个算法

我们说过，很多问题都是有难度的。那么你如何拿出一个算法呢？这里有几种好用的策略。

- **使用具体的例子** 拿到一个问题时，不要只是坐在椅子上在脑子里推演。站起来，走到白板前，勾勒出一个例子。比如说，如果你想找出合并两个有序数组的办法，那就以两个具体的数组为例，比如`{1, 5, 8, 9}`和`{3, 5, 7, 10, 12}`，然后走一遍你的合并过程。用具体的值（不是 `a1` 和 `a2` 这样的变量），并且要避免特殊的情况（比如没有共同元素的数组）。
- **优化蛮干** 一开始不要羞于使用蛮干或幼稚的方案。这样你能有一个良好的，可以优化的基础。如果蛮干很慢，想想它为什么慢？算法中的哪一步是最大的时间结点？重点优化它们。
- **解决基本情况** 有时候，从小处开始解决问题比较容易。试着在值为 0 时求教，然后为 1，为 2，等等以此类推。能看到某种模式吗？或者你能用前面这些答案做出更大输入值的答案？比如说，如果你要算出`{a, b, c}`的所有子集，可能可以用到`{a, b}`的子集。
- **考虑类似的问题** 如果这个问题跟你以前听过的其他问题类似，看看能否用类似的方式解决这个问题。解决技术问题的练习越多，越容易拿出一个算法。
- **简化和调整** 面试中给出的问题会有一些限定条件，比如输入的大小、输入的类型、范围，或其他因素。可以尝试以某种方式调整或简化这些问题，看能否解决修改后的版本。
- **记录你的见解** 在你解决问题时，可能会发现问题的关键“点”。比如说，你在树中找一个值时，意识到它一定在左侧子树的右边。记住那个线索。

不管做什么，都要积极地解决问题。不要因为痛苦的煎熬而丧失勇气，更不要放弃。面试官想要看到披荆斩棘、百折不挠的你。

16.6 附加题

16.19 设计一个算法并编写代码，找出等式 $a^3 + b^3 = c^3 + d^3$ 的所有解，其中 a、b、c、d 都是小于 1000 的正整型值。如果你愿意，可以只输出“有意思”的解。即你可以忽略 $x^3 + y^3 = x^3 + y^3$ 这种形式的解，以及哪种是其他解经过简单排列后的解（交换了等式的左侧和右侧，交换了 a 和 b，交换了 c 和 d）。比如说，如果输出所有小于 20 的解，你可以只输出 $2^3 + 16^3 = 9^3 + 15^3$ 和 $1^3 + 12^3 = 9^3 + 10^3$。

248 页

16.20 给出一个字符串，输出那个字符串的所有排列组合（假定这个字符串中没有重复的字符）。

252 页

16.21 在一群人中，所有人互相都认识，但其中也有不认识其他人的那种人，这种人被称为“名人”。给你一个函数 `knows(a,b)`，可以告诉你 `a` 是否认识 `b`。设计一个算法找出名人（如果有的话）。

简单起见，假定每个人都有一个标签，范围从 `0` 到 `N-1`。你需要实现一个函数 `int findCelebrity(int N)`。注意看：

(1) 最多只能有一个名人（基于名人的定义）；

(2) 给出的函数是找到谁认识谁的唯一办法。

254 页

16.22 有一个 $N \times N$ 的字母表，一个有效单词的列表（可以是你希望的任何格式）。可以从任何一个字符开始，然后上下左右移动来形成一个单词。单词不必在一条直线上（比如下面这个字母表中形成的单词 `PACKING`）。字母不能在单词中重复使用，所以 `GOING`（在下面的字母表中）不成立，因为它重用了 `G`。设计一个算法，并写代码输出所有有效的单词。

```
L I G O
E P N I
N A C K
S M A R
```

256 页

16.23 给定一个整形数组（既有正值又有负值），找到和值最大的连续序列（只返回和值）。比如：

```
输入：2, -8, 3, -2, 4, -10
输出： 5 (即 {3, -2, 4})
```

258 页

16.7 参考答案

所有参考答案都是用 Java 实现的。如果你不懂 Java，没关系。我们会尽量让代码不受 Java 语法复杂性的影响，这样你就可以专注于最主要的算法。

16.1 给定一个已排好顺序的正整值数组，末尾有一个空位（0），按顺序插入一个元素。

211 页

我们可以将数组想象成下面这种（最后有个空位）：

```
1 4 7 8 9 _
```

如果需要插入一个元素，比如说 6，则不能只是把它插在后面，应该按顺序插入。

```
1 4 6 7 8 9
```

这需要“偏移”下面所有元素，给 6 腾出空间后插入它。

解决这个问题有两种办法。

方法 1：从后面偏移，然后插入

第一种方式是移动所有元素，然后插入值 x。然而一定要小心，别在插入时覆盖了原来的值。

与其从前面移动，我们不如从后往前移动。

```
1 4 7 8 9 _
```

先把 9 复制到空位上，然后 8 挪到 9 之前所在的位置，然后 7 到 8 之前的位置，依此类推。等我们找到 x 的合适位置时，就停下来插入 x。

```
1   boolean insert(int[] array,int x) {
2      /* 确保输入有效。*/
3      if (array[array.length - 1] != 0 || x <= 0) {
4        return false;
5      }
6
7      /* 从最后一个非空的元素开始，向左移动并逐一复制各元素。找到 x 的正确位置时，或者到达数组的
         第一个元素时停止*/
8      int index = array.length - 2; // 倒数第二个
9      while (index >= 0 && array[index] > x) {
10        array[index + 1] = array[index]; // 移动一个位置
11        index = index - 1; // 到下一个元素
12     }
13
14     /* 当上面的循环停止时插入元素。*/
15     array[index + 1] = x;
```

```
16
17     return true;
18  }
```

如果元素能够插入，返回 `true`；如果有错，返回 `false`。

方法 2：交换元素向前移动

此外，我们也可以循环移动数组。对于数组中前面的元素（那些小于 x 的），什么都不用做，它们不会被移动。

然而找到插入 x 的位置时，把 x 跟数组中的当前元素互换，现在 x 就等于数组中的原有元素了。

到下一个元素时，我们再把 x 跟那个值交换，对数组中的后续元素做这个操作，直到到达最后。

```
insert 6 into 2, 3, 7, 8, 9, _
set x = 6
start i at A[0]
move i to A[1]
move i to A[2]
swap A[2] and x.
    A = {2, 3, 6, 8, 9, _}
    x = 7
swap A[3] and x.
    A = {2, 3, 6, 7, 9, _}
    x = 8
swap A[4] and x.
    A = {2, 3, 6, 7, 8, _}
    x = 9
swap A[5] and x.
    A = {2, 3, 6, 7, 8, 9}
    x =_
```

下面这段代码实现了这个算法：

```
1   boolean insert(int[] array,int x) {
2      /* 确保输入有效*/
3      if (array[array.length - 1] != 0 || x <= 0) {
4         return false;
5      }
6
7      for (int i = 0; i < array.length; i++) {
8         if (x < array[i] || array[i] == 0) {
9            /* 交换 x 和 array[i].*/
10           int temp = array[i];
11           array[i] = x;
12           x = temp;
13        }
14     }
15
16     return true;
17  }
```

注意，一旦第 8 行的 `if` 语句变成 `true`，它就会一直为 `true`。

两个算法的时长都是 $O(N)$。

16.2　按逆序排列数组中的元素（不创建新数组）。

211 页

乍一看，我们可能想要创建第二个数组，按顺序遍历元素，然后按相反的顺序插到新数组中。可惜，题目要求不能创建第二个数组。

随机看一个例子。

```
原始数组：0, 1, 2, 3, 4, 5, 6
逆序后的数组：6, 5, 4, 3, 2, 1, 0
```

可能你也注意到了，数组在逆序排列时，`0` 被放到了 `6` 的位置上，而 `6` 被放到了 `0` 的位置上。同样，`5` 和 `1` 也互换了位置，也就是说我们是在交换值的位置。

无需创建第二个数组，就可以遍历数组，把左边的值跟右边相应的值交换。因为遍历数组左半边时右半边就已经处理到了，所以只需要遍历左半边。

```
1   void reverse(int[] array) {
2      int midpoint = array.length / 2;
3      for (int i = 0; i < midpoint; i++) {
4         /* 得到右侧对应值的索引*/
5         int otherside = array.length - 1 - i;
6
7         /* 交换左侧和右侧的值*/
8         int temp = array[otherside];
9         array[otherside] = array[i];
10        array[i] = temp;
11     }
12  }
```

对第 2 行和第 5 行的数学计算一定要非常小心，这是你在面试中应该再三检查的事情。

两个算法的时长都是 $O(N)$。

16.3　假定有两个字符串链表（A 和 B），其中包含的字符都是唯一的，写一个程序确定 A 是否为 B 的子集，也就是检查 A 的所有元素是否都在 B 中。

211 页

题目中说这两个列表中包含的字符串没有重复的，所以只需要检查一个列表中是否包含另一个列表中的所有字符串就好了。

方法 1：蛮力法

我们可以用“蛮力”解决这个问题，即逐一检查 A 中的每个元素是否在 B 中。

一旦在 A 中发现了不在 B 中的元素，就可以返回 false 了，因为那说明 A 不是 B 的子集。如果到达 A 的末尾时还没返回，就说明 A 中所有元素在 B 中都有。返回 true。

```
1  boolean isSubsetBruteForce(String[] bigger,String[] smaller) {
2     for (String s : smaller) {
3        boolean found = false;
4        for (String b : bigger) {
5           if ( s.equals(b)) { // 找到元素
6              found = true;
7              break;
8           }
9        }
10       if (!found) { // s 不是 found -> 不是子集
11          return false;
12       }
13    }
14    return true; // 找到了所有元素
15 }
```

这个算法的时长是 $O(a \times b)$，其中 a 是 A 的长度，b 是 B 的长度。

方法 2：散列表

前面那种办法之所以那么慢，是因为我们在处理每个元素时都要遍历 B。如果能直接查找一个元素是否在 B 中是不是更好？

可以的！用散列表就可以实现。为 B 中所有元素构建一个散列表，然后当我们想要查找某个元素是否在 B 中时，只要使用这个散列表就可以了。

```
1   boolean isSubset(String[] bigger,String[] smaller) {
2      Hashtable<String,Boolean> hase =
3         new Hashtable<String,Boolean>();
4
5      /*记录较大列表中的所有元素*/
6      for (String b:bigger) {
7         hash.put(b,true);
8      }
9
10     /*检查散列表中是否包含所有字符串。*/
11     for (String s : smaller) {
12        if ( !hash.containsKey(s) || hash.get(s) != true) {
13           return false;
14        }
15     }
16    return true;
17 }
```

这个算法的时长是 $O(a+b)$，其中 a 是 A 的长度，b 是 B 的长度，此外还需要 $O(b)$的额外内存来存放散列表。

16.4 假定有一个含有销售数据的二维数组，第一列是产品 ID，第二列是数量。写一个函数，以这个数据列表为输入，返回一个新的二维数组，其中是每个产品 ID 及其总销量。

示例：

输入：

```
211,4
262,3
211,5
216,6
```

输出：

```
211,9
262,3
216,6
```

212 页

这个方法要输出包含产品 ID 和它们总销量的列表，可以用散列表以简单直接的方式实现。

循环遍历包含`(productID, quantity)`对的列表，对于每个 `productID`，都增加它在散列表中的记录，如果还没有添加到散列表中，就把它插进去，最后再把散列表转换成数组。

```
1   int[][] totalSales(int[][] data) {
2      Hashtable<Integer, Integer> hash =
3         new Hashtable<Integer, Integer>();
4
5      /* 计算每个产品的总销量。*/
6      for (int i = 0; i < data.length; i++) {
7         int productId = data[i][0];
8         int quantity = data[i][1];
9         if (hash.containsKey(productId)) {
10           quantity = quantity + hash.get(productId);
11        }
12        hash.put(productId, quantity);
13     }
14
15     /*将散列表转换成数组。*/
16     int[][] totals = new int[hash.keySet().size()][2];
17     int index = 0;
18     for (int key : hash.keySet()) {
19        totals[index][0] = key;
20        totals[index][1] = hash.get(kye);
21        index = index + 1;
22     }
23     return totals;
24  }
```

如果你不了解 keySet 和 containsKey 之类的特殊命令，不用担心。面试官也不关心这些。重点是你知道如何将一种解法变成貌似可行的代码。

这个算法的时长是 $O(N)$，其中 N 是输入中的行数。

16.5 往二叉查找树里插入一个元素（按顺序），假定这个二叉查找树里包含的是整数。

213 页

按照二叉查找树的定义，这是一个简单直接的问题。

在二叉查找树中，较小值放在节点左侧，较大值在节点右侧。

解决这个问题最容易的解法是递归。从根节点开始，与要插入的值 x 比较。如果 x 小于根节点，则在 root.left 上调用 insert。如果 x 大于根节点，则在右侧调用。重复这个过程，直到没有左或右子节点，然后在那里插入 x。

```
1  boolean insert(TreeNoderoot,intdata) {
2     if (root == null) return false; // 失败
3
4     if (data <= root.data) {
5        if (root.left == null) { // 找到位置
6           root.left = new TreeNode(data); // 插入
7        } else {
8           return insert(root.left, data); // 递归
9        }
10    } else {
11       if (root.right == null) { // 找到位置
12          root.right = new TreeNode(data); // 插入
13       } else {
14          return insert(root.right,data); // 递归
15       }
16    }
17    return true; // 成功
18 }
```

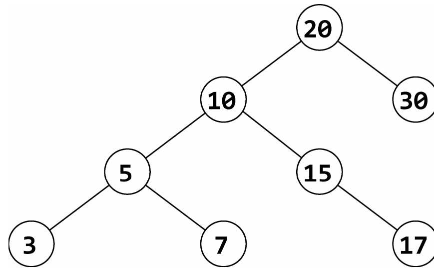
插入节点所需的时间取决于树的高度。如果这棵树相对平衡，它的高度应该是 $O(\log N)$，其中 N 是树中的节点数。然而，如果这棵树非常不平衡（比如基本上是所有节点都在同一侧的一条从上到下的直线），高度最高可能是 N。

16.6 给定一个包含整数的二叉查找树，计算所有数的和值。

213 页

如果解题的切入点正确，有些问题简单得超乎想象。在这个问题上，“正确的”切入点就是递归。

假定我们想要计算这样一棵树中所有节点的和值：



可以遍历这棵树，将它收到一个数组里，然后计算那些值的和值。实际上没必要那么复杂。

更简单的方式是将这个问题分解成子问题。整棵树的和值是左侧子树的和值加上右侧子树的和值再加上根的和值。

```
sum(tree_at_20) =
      sum(tree_at_10)
    + sum(tree_at_30)
    + value_at_node_20
```

求节点 10 上的和值也可以用它的子问题定义。

```
sum(tree_at_10) =
      sum(tree_at_5)
    + sum(tree_at_15)
    + value_at_node_10
```

几乎可以把这个直接变成代码。

```
1   int sum(TreeNoderoot){
2      if (root == null) {
3         return 0;
4      }
5      return root.data + sum(root.left) + sum(root.right);
6   }
```

如果到了路径的终点（一个 `null` 节点），则返回 `0`，或者也可以这样：

```
1   int sum(TreeNoderoot) {
2      if (root == null) {
3         return 0;
4      }
5
6      int total = root.data;
7      if (root.left != null) {
8         total += sum(root.left);
9      }
10     if (root.right != null) {
11        total += sum(root.right);
```

```
12      }
13      return total;
14  }
```

不管用哪段代码，运行时长都是 O(*N*)，其中 N 是树中节点的数量。

运行时长之所以是 O(*N*)，是因为 sum 在树中的每个节点上只会调用一次。如果对 sum 的调用是 O(*N*)，那运行时长应该也是 O(*N*)。

16.7 向已排序链表中插入一个节点（按顺序）。（别忘了新元素在链表头或尾部时会发生什么！）

213 页

要往链表中插入一个数值，先要找到正确的插入位置，然后还要插入它。

找出如何在链表前面插入一个节点需要一点儿技巧。

假设我们可以调用这样一个 insertInOrder 方法，并且它（在这个特例中）需要在列表前面插入 n：

```
void insertInOrder(LinkedListNode nd, int value)
```

只是插入节点 n 并把 n.next 指向 nd 还不够。使用链表的人都不知道它真正的表头已经从 nd 变成了 n，他们只有对 nd 的引用。

因此在 insert 方法中需要返回链表的新表头。多数情况下，表头还和你调用 insert 之前的一样，然而有时它会变，并且你需要将这一变化告知调用者。

```
1   LinkedListNode insert(LinkedListNode head,int data){
2      /* 创建新节点*/
3      LinkedListNode node = new LinkedListNode(data);
4
5      /* 如果被插入到链表前面，则插入节点并返回新的表头。*/
6      if (head == null || data < head.data) {
7         node.next = head;
8         return node;
9      }
10
11     /* 通过遍历链表找到插入节点的正确位置 (按顺序) 。*/
12     LinkedListNode current = head;
13     while (current.next != null && data > current.next.data) {
14        current = current.next;
15     }
16
17     /* 插入节点。*/
18     node.next = current.next;
19     current.next = node;
```

```
20
21    /* 返回旧表头。它还没动。*/
22    return head;
23  }
```

这个算法的时长是 $O(N)$，其中 N 是节点数。

16.8　对一个只包含 0 和 1 的链表排序，也就是将链表中的所有 0 都放到 1 之前。

213 页

这个问题有很多种解法：

方法 1：建两个链表

最简单的办法是建一个“0 值链表”和一个“1 值链表”，然后在尾部联接。

```
1   LinkedListNode sort(LinkedListNode head) {
2     LinkedListNode zeroHead = null;
3     LinkedListNode zeroTail = null;
4     LinkedListNode oneHead = null;
5     LinkedListNode oneTail = null;
6
7     LinkedListNode n = head;
8     while (n != null) {
9        LinkedListNode next = n.next;
10       n.next = null;
11       if (n.data == 0) {
12          /* 添加到 0 值链表的尾部。*/
13          if (zeroHead == null) {
14             zeroHead = n;
15          } else {
16             zeroTail.next = n;
17          }
18          zeroTail = n;
19       } else {
20          /* 添加到 1 值链表的尾部。*/
21          if (oneHead == null) {
22             oneHead = n;
23          } else {
24             oneTail.next = n;
25          }
26          oneTail = n;
27       }
28        n = next;
29     }
30
31     /* 合并链表并返回。*/
32     if (zeroTail == null) {
33        oneTail.next = null;
34        return oneTail;
```

```
35     } else {
36        zeroTail.next = oneHead;
37        return zeroHead;
38     }
39  }
```

注意看，因为链表的表头可能会变，所以得返回它。

方法 2：在左侧和右侧增长

前面那个解法代码很长，因为需要跟踪两个不同链表的前面和后面，因此需要不断地更新四个不同的变量。

实际上并不需要四个变量，只要把所有的 0 放在所有的 1 前面，并不需要按原来的顺序存放这些节点。

因此，只要跟踪（新）链表的头和尾。当拿到一个新的 0 时，把它插在前面就好了。拿到新的 1 时，把它放到后面，这样所有的 0 就会在所有的 1 前面了。

```
1  LinkedListNode sort(LinkedListNode n) {
2     LinkedListNode head = n;
3     LinkedListNode tail = n;
4     n = n.next; //从第二个元素开始
5
6     while (n != null) {
7        LinkedListNode next = n.next;
8        if (n.data == 0) { // 0 -> 加到前面
9           n.next = head;
10          head = n;
11       } else { // 1 -> 加到尾部
12          tail.next = n;
13          tail = n;
14       }
15       n = next;
16    }
17    tail.next = null; // 确保尾部没有指向任何位置
18
19    return head;
20 }
```

还是需要返回新的头部，因为它还是可能会变。

方法 3：对 0 计数

实际上题目没要求用给定的对象，如果移动那些值，而不是节点，也能满足问题的要求。

因此可以只是遍历一次链表，数出 0 的个数。然后将前 k 个值设为 0，并将其他的值设为 1。

```
1  void sort(LinkedListNode head) {
2     int k = 0;
3
```

```
4      /*对 0 计数*/
5      LinkedListNode n = head;
6      while (n != null) {
7         if (n.data == 0) {
8            k = k + 1;
9         }
10        n = n.next;
11     }
12
13     /* 将前面 k 个值设为 0。*/
14     n = head;
15     while (n != null) {
16        if (k > 0)  {
17           n.data = 0;
18           k = k - 1;
19        } else {
20           n.data = 1;
21        }
22        n = n.next;
23     }
24  }
```

方法 4：交换值

因为只需要移动这些数值，所以也可以遍历链表，发现 0 和 1 时交换它们的位置。

这种办法需要两个指针，p 和 q。指针 p 指向 1，q 指向 0。当它们找到各自的值时就互相交换。

(1) p 从表头开始

```
0->0->0->1->1->0->1->0->1->0
p
```

(2) 将 p 移动到第一个 1 上

```
0->0->0->1->1->0->1->0->1->0
         p
```

(3) 将 q 放在 p.next 的位置上

```
0->0->0->1->1->0->1->0->1->0
         p  q
```

(4) 将 q 移动到下一个 0 上

```
0->0->0->1->1->0->1->0->1->0
         p     q
```

(5) 交换 p 和 q 的值

```
0->0->0->0->1->1->1->0->1->0
         p     q
```

(6) 重复步骤 4：

```
0->0->0->0->1->1->1->0->1->0
            p  q                //将 p 移动到下一个 1 上
0->0->0->0->1->1->1->0->1->0
            p        q          //将 q 移动到下一个 0 上
0->0->0->0->0->1->1->1->1->0
            p        q          //交换
0->0->0->0->0->1->1->1->1->0
               p        q       //将 p 移动到下一个 1 上
0->0->0->0->0->1->1->1->1->0
               p           q    //将 q 移动到下一个 0 上
0->0->0->0->0->0->1->1->1->1
               p           q    //交换
```

换句话说，p 总是指向第一个 1，q 总是指向第一个错位的 0（p 之后的第一个 0）。只要 q 找到 0，就知道这个 0 错位了。交换它跟值的 p，然后将 p 挪到下一个节点。

这个办法对有些人来说可能不太直观，但在辅助函数的帮助下，它的代码相当短。

```
1   void sort(LinkedListNode head) {
2      LinkedListNode p = nextNodeWithVal(head,1);  //找到第一个 1
3      LinkedListNode q = nextNodeWithVal(p.next,0); //找到下一个 0
4   
5      while (p != null && q != null) {
6         q.data = 1; // 交换
7         p.data = 0;
8         p = nextNodeWithVal(p,1); //找到下一个 1
9         q = nextNodeWithVal(q,0); //找到下一个 0
10     }
11  }
12  
13  LinkedListNode nextNodeWithVal(LinkedListNode n,int data) {
14     while (n != null && n.data != data) {
15        n = n.next;
16     }
17     return n;
18  }
```

这些就是解决这个问题的四种解法，但还有很多其他解法，这四种解法的时长全都是 $O(N)$。

16.9　写一个函数，以栈为输入，然后返回一个元素顺序相反的栈。

214 页

这个问题最简单直接的解法就是创建一个新栈，然后把第一个栈中的元素弹出来压到第二个栈里。这会把原来那个栈顶端的元素压到新栈的底部。

```
1   Stack<Integer> reverse(Stack<Integer> stack) {
2      Stack<Integer> reversed = new Stack<Integer>();
```

```
3     while (!stack.isEmpty()) {
4         reversed.push(stack.pop());
5     }
6     return reversed;
7  }
```

这种解法唯一的问题是处理完后最初那个栈完全空了。如果这样不行（问一下面试官），可以再用一个栈保存所有弹出的值。

把弹出值同时压入临时栈和反序栈中。（这些栈以相同的顺序（反序）保存相同的元素。）原始栈中的所有元素全部弹出后，再把临时栈中的元素压回到原始栈中。

```
1  Stack<Integer> reverseWithoutDestroying(Stack<Integer> stack) {
2     Stack<Integer> reversed = new Stack<Integer>();
3     Stack<Integer> temp = new Stack<Integer>();
4     while (!stack.isEmpty()) {
5        int x = stack.pop();
6        reversed.push(x);
7        temp.push(x);
8     }
9
10    while (!temp.isEmpty()) {
11       stack.push(temp.pop());
12    }
13
14    return reversed;
15 }
```

两种办法的时长都是 $O(N)$。第二种办法要过两轮，但常量不会影响大 O 的值。有些人可能会觉得奇怪，但请记住：大 O 不是某事所需的真正秒数，它表示的是随着输入变得越来越长时间会如何变化（在此例中是线性的）。

16.10　写一个函数，从栈中移除所有偶数。你应该返回原来那个栈，而不是新建一个。

214 页

对于这个问题，可以采用前面那个问题的第二种方式：两次逆序可以把元素按原顺序放回去。

可以只是把元素逐个弹出栈。如果弹出的元素是奇数（即非偶数），则把它压入一个新的临时栈中。然后等做完后，再把它们压回原始栈中。

```
1  void removeEvens(Stack<Integer> stack) {
2     Stack<Integer> temp = new Stack<Integer>();
3     while (!stack.isEmpty()) {
4        int x = stack.pop();
5        /*把奇数压入新栈*/
6        if ( x % 2 != 0) {
```

```
7           temp.push(x);
8        }
9     }
10
11    /*将奇数返回到原始栈*/
12    while (!temp.isEmpty()) {
13       stack.push(temp.pop());
14    }
15 }
```

这个算法的时长是 $O(N)$。注意观察，因为必须把每个元素都过一遍，所以这个问题不可能有更快的解法了。

16.11　写一个函数检查两个队列是否相同（以相同顺序排列的相同值）。修改/销毁这两个队列也没关系。

214 页

可以修改两个队列，这应该能说明只需要做那个就行了。

可以不断地移除链表前面的每个元素并进行比较。如果值相等，马上返回 `false`。

当列表变成空的时会发生什么？那要看情况。如果两个列表都是空的，那这两个链表相同（还什么都没失败）。然而如果只有一个列表是空的，而另一个不是，那这两个列表的大小是不同的。毕竟我们是按相同的顺序移除元素的。

```
1  boolean isEqual(Queue<Integer> one,Queue<Integer> two) {
2     /* 逐个移除元素，检查它们是否相等*/
3     while (!one.isEmpty() && !two.isEmpty()) {
4        int oneHead = one.remove();
5        int twoHead = two.remove();
6        if (oneHead != twoHead) {
7           return false;
8        }
9     }
10
11    /* 我们很高兴看到有一个列表已经空了。如果另外一个列表里面还有元素，那这两个列表大小肯定不同。*/
12    if (!one.isEmpty() || !two.isEmpty()) {
13       return false;
14    }
15
16    return true;
17 }
```

这个算法的时长是 $O(N)$，其中 N 是较小列表的长度。为什么是较小列表呢？因为只要有列表为空就退出了。所以是较小列表先满足这个条件。比较大的列表多长都没关系；它不会影响运行时长。

16.12 写一个函数，从队列中移除第 13 个元素（但要保证其他元素还按原来的顺序排列）。

214 页

这个问题的解法取决于你假定队列数据结构支持什么。

如果可以访问 Node 类，那这个相当容易。只要遍历节点，然后到第 13 个时删掉它就行了。

然而如果它是一个真正的 Queue 类，就不一定能这样访问节点。可能只有添加（向列表尾部）和移除（从列表头部）方法。

可以再创建一个列表对象，但其实也没必要。因为如果不断地从头部移除元素，再把它们添加到尾部，最终还能得到一个一模一样的列表。

要移除第 13 个元素，可以只是移除每个元素再重新添加它，只是添加时要跳过第 13 个元素。

在实现这段代码时用了变量 k，而不是把 13 硬编码在代码中。这通常是个不错的编码实践。

```
1   boolean remove(Queue<Integer> queue,int k) {
2      if (k<0 || k>=queue.size()) {
3         return false;
4      }
5      int size = queue.size();
6      for (int i = 0; i < size; i++) {
7         int head = queue.remove();
8         if (i != k ) { /*除第 k 个之外的所有元素从前移除并添加到后面*/
9            queue.add(head);
10        }
11     }
12     return true;
13  }
```

这个算法的时长是 $O(N)$，其中 N 是节点数。

16.13 给定两个已排好序的数组，写一个函数，按顺序将其中的元素合并到一个新数组中。

215 页

解决这个问题最有效的办法是利用数组已经排好序的事实。可以不断地合并连续元素，直到到达两个数组的结尾处。要维护两个指针，指向每个数组中的位置，这样就可以轻松地切换到下一个数组。

来看两个数组的例子。

```
A: 1 5 8 9      B: 2 4 9 10 12
```

在两个数组的头部分别放两个指针 p 和 q：

```
A: 1 5 8 9      B: 2 4 9 10 12
   p               q
```

A[p]小于 B[q]，所以我们把 A[p]放到结果数组中，然后将 p 移动到下一个值上。

```
A: 1 5 8 9      B: 2 4 9 10 12     Result: 1
     p             q                       i
```

再次比较 A[p]和 B[q]，把比较小的元素放到结果数组中。还要追踪在结果数组中处于什么位置，重复这一过程，直到两个数组都做好。

```
A: 1 5 8 9      B: 2 4 9 10 12      Result: 1 2
     p               q                        i
A: 1 5 8 9      B: 2 4 9 10 12      Result: 1 2 4
     p                 q                        i
A: 1 5 8 9      B: 2 4 9 10 12      Result: 1 2 4 5
       p               q                          i
A: 1 5 8 9      B: 2 4 9 10 12      Result: 1 2 4 5 8
         p             q                            i
A: 1 5 8 9      B: 2 4 9 10 12      Result: 1 2 4 5 8 9
                       q                              i
A: 1 5 8 9      B: 2 4 9 10 12      Result: 1 2 4 5 8 9 9
                          q                             i
A: 1 5 8 9      B: 2 4 9 10 12      Result: 1 2 4 5 8 9 9 10
                             q                            i
A: 1 5 8 9      B: 2 4 9 10 12      Result: 1 2 4 5 8 9 9 10 12
                                                             i
```

面试时，做这样一个详细的例子可以减少你犯错误的机会。

```
1   int[] mergeIntoNew(int[] A,int[] B) {
2      int p = 0;
3      int q = 0;
4      int index = 0;
5      int[] merged = new int[A.length + B.length];
6
7      while (p < A.length || q < B.length) {
8         if (q >= B.length || A[p] <= B[q]) {
9            merged[index] = A[p];
10           p = p + 1;
11        } else {
12           merged[index] = B[q];
13           q = q + 1;
14        }
15        index = index + 1;
16     }
17
18     return merged;
19  }
```

如果你想减少指针的数量，可以去掉变量 index。它总是等于 p + q。

这段代码的时长是 $O(M+N)$，其中 M 是第一个数组的长度，N 是第二个数组的长度。

16.14　实现插入排序。

215 页

插入排序是通过遍历数组完成的，按顺序在元素的左侧插入每个元素。

可以将这个做成两个不同的函数，那样更清晰。

第一个函数执行全局算法：挑选一个元素，按顺序插入，挑选下一个元素，依此类推。

```
1  void insertionSort(int[] array) {
2     /*从左侧开始挑选元素并按顺序查到左侧*/
3     for (int i = 1; i < array.length; i++) {
4        insertInOrder(array, i);
5     }
6  }
```

注意看 `for` 循环，是从 1 而不是 0 开始的。这是因为第 0 个元素自身永远不可能是乱序的（单个元素的子数组总是排好序的）。

现在只需要实现一个方法，接受元素 `A[k]`，并将其按顺序插入到它左侧的元素中（假定已经排好序了）。

要按顺序插入 `A[k]`，我们需要逐一移动每个元素，直到为这个元素找到正确的插入点。

```
1  void insertInOrder(int[] array,int index) {
2     int x = array[index]; //把元素存为临时值
3     index = index - 1;
4     while (index >= 0 && array[index] > x) {
5        array[index + 1] = array[index]; //移动一个位置
6        index = index - 1;
7     }
8     array[index + 1] = x; // 插入元素
9  }
```

这个算法的时长是 $O(N^2)$。

16.15　实现二分查找：即给定一个已排序的整形数组和一个值，找到那个值的位置。

216 页

二分查找要不断地把数组“对半分”成子数组。在第一轮中，将 x 跟中间位置比较，确定 x 是在左半边还是右半边。然后对新的子数组重复这一步骤：x 是在它（新的子数组）的左半边还是右半边？

对于这个问题，既可以用递归实现，也可以用循环（非递归）。先实现递归的解法，因为对大多数人来说它更直观。

```
1   int search(int[] array,int x) {
2      return search(array, x, 0, array.length - 1);
3   }
4
5   int search(int[] array,int x,int left,int right) {
6      if (right < left) { // 没找到
7         return -1;
8      }
9
10     int middle = (right + left) / 2;
11     if (x == array[middle]) {
12        return middle;
13     } else if (x < array[middle]) { // x 在左侧
14        return search(array, x, left, middle - 1);
15     } else { //x 在右侧
16        return search(array, x, middle + 1, right);
17     }
18  }
```

使用循环的解法非常类似：

```
1   int searrch(int[] array,int x) {
2      int left = 0;
3      int right = array.length - 1;
4      while (left <- right) {
5         int middle = (right + left) / 2;
6         if ( x == array[middle]) {
7            return middle;
8         }
9         if (x < array[middle]) {
10           right = middle - 1;
11        } else {
12           left = middle + 1;
13        }
14     }
15     return -1;
16  }
```

考虑一下从递归变成循环的实现时会有什么样的差异，这对你是个很好的锻炼机会。比如说，使用递归时第 6 行的检查会出现什么情况？

16.16　给你一个已排序的整型数组，然后旋转一下。其中的元素都是唯一的。找到其中的最小值。比如说，数组可能是 6，8，9，11，15，20，3，4，5。最小值很明显是 3。

216 页

使用蛮力的解法是遍历数组，找到最小值。但估计这不是面试官想要的解法，因为它用不到

排序信息。

要找出更优的方案，可能想要使用题目中给出的信息，这个数组是“有序的”，但被旋转了。

因为这个数组在某种程度上是有序的，可以考虑用一些二分查找中的概念，二分查找要不断找中点。

在这个问题里，中点能说明什么？就其本身而言，中点是 15 也说明不了什么。然而，如果已知其左侧是 6，右侧是 5，那我们能得出一些结论：既然左侧>右侧，说明这个数组是无序的；但同时左侧<中点，则说明左侧是有序的，但右侧不是。

```
6, _, _, _, 15, _, _, _, 5
```

通过对上面数组的检查，可以确定拐点（最小的元素）在右半边，现在我们的问题被分成两半了。

要找到最小的元素，只要递归就可以了

```
20, _, _, 5
20, 3
3
```

这个可以用递归实现，发现左侧比右侧小时就可以停止了。这表明这部分数组是有序的，因此最左面就是最小的元素。

```
1   int findMin(int[] array,int left,int right) {
2      /*数据项是有序的。因此左边必须是最小的。*/
3      if (array[left] <= array[right]) {
4         return left;
5      }
6
7      /* 在左侧找最小的元素。*/
8      int middle = (right + left) / 2;
9      if (array[left] > array[middle]) {
10        return findMin(array,left,middle);
11     } else { //中间元素 > 右侧元素
12        return findMin(array,middle + 1,right);
13     }
14  }
```

此外，也可以用一个 while 循环实现这个算法：

```
1   int findMin(int[] array) {
2      int left = 0;
3      int right = array.length - 1;
4      while (array[left] > array[right]) {
5         int middle = (left + right) / 2;
6         if (array[left] > array[middle]) {
7            right = middle;
8         } else {
```

```
9           left = middle + 1;
10        }
11     }
12     return left;
13  }
```

在这样的问题中，一定要非常小心地确定你的终止和递归条件。想想你为什么让 `left = middle + 1`(为什么+1?)而不是 `right = middle`，那些地方很容易出错。

16.17 使用深度优先搜索，检查一棵树中是否有某个值。

217 页

深度优先搜索要检查值 v 是否等于当前节点的值。如果不等，则逐一搜索节点的子节点。

深度优先搜索（DFS）和广度优先搜索（BFS）的不同之处在于，在 DFS 中，会在转向节点的其他任何子节点之前把这个节点的整个子树搜索完。也就是说 `node.child[0]`的所有子节点的搜索会在你查找 `node.child[1]`之前完成。

可以用递归实现：

```
1  boolean depthFirstSearch(TreeNode node,int x) {
2     if (node == null) {
3        return false;
4     } else if (node.data == x) {
5        return true;
6     } else {
7        return depthFirstSearch(node.left, x) ||
8               depthFirstSearch(node.right, x);
9     }
10 }
```

因为这是一棵树，所以不必担心它会死循环，那就是说不用担心在节点的子节点上的遍历，不用担心在“孙子”节点上的遍历，也不用担心会突然跳回到最初的节点，结果困在一个死循环里。树会特别禁止这样的环路。

如果不是树，而是在图中，就要用一个 `isVisited` 标记来标出已经访问过的节点。

16.18 编写在二叉树上进行广度优先搜索的伪码（尽可能详细）。

217 页

要执行广度优先搜索，要一层一层地搜索节点。也就是说要先搜索每个节点的子节点，然后再搜索这些子节点的子节点。

尽管广度优先搜索的概念简单直白（只要逐层地搜索节点的子节点），但实现却不是那么直观。其中的诀窍是记住我们需要用到队列。

可能你还记得，队列这种数据结构可以在一边添加数据，在另一边移除。它是一个“先进先出”（FIFO）的数据结构，这样就基本上可以将节点做成“稍后处理的”。

在 BFS 中，“访问”一个节点时会将正在查找的`(x)`跟当前值进行比较。如果匹配，则结束查找并马上返回 `true`。否则将节点的子节点添加到队列的尾部，然后继续，从队列的另一边拉一个节点搜索。

```
1   boolean searchBFS(TreeNode root,int x) {
2      Queue<TreeNode> queue = new LinkedList<TreeNode>();
3      return searchBFS(root, x, queue);
4   }
5
6   boolean searchBFS(TreeNode root,int x,Queue<TreeNode> queue) {
7      queue.add(root);
8      while (!queue.isEmpty()) {
9         TreeNode node = queue.remove();
10        if (node.data == x) {
11           return true;
12        }
13        if (node.left != null) {
14           queue.add(node.left);
15        }
16        if (node.right != null) {
17           queue.add(node.right);
18        }
19     }
20     return false;
21  }
```

因为这是一棵树，所以不用担心会出现环路。然而如果不是这种情况，就需要用 `isVisited` 来标记已经访问过的节点，以免重复访问。

广度优先搜索的时长是 $O(N)$，其中 N 是图（或树）中的节点数，这是因为可能要搜索所有的节点。

16.19 设计一个算法并编写代码，找出等式 $a^3+b^3=c^3+d^3$ 的所有解，其中 a、b、c、d 都是小于 1000 的正整型值。如果你愿意，可以只输出“有意思”的解。即你可以忽略 $x^3+y^3=x^3+y^3$ 这种形式的解，以及那种是其他解经过简单排列后的解（交换了等式的左侧和右侧，交换了 a 和 b，交换了 c 和 d）。比如说，如果输出所有小于 20 的解，你可以只输出 $2^3+16^3=9^3+15^3$ 和 $1^3+12^3=9^3+10^3$。

227 页

可以先实现一种很幼稚的解法，为 a、b、c、d 循环遍历所有可能的值。相等的时候输出这一组值。

```
1   void cubes(int max) {
2      for (int a=0; a < max; a++) {
3         int acubed = a * a * a;
4         for (int b = 0; b < max; b++) {
5            int bcubed = b * b * b;
6            for (int c = 0; c < max; c++) {
7               int ccubed = c * c * c;
8               for (int d = 0; d < max; d++) {
9                  int dcubed = d * d * d;
10                 if (acubed + bcubed == ccubed + dcubed) {
11                    String solution = a + "," + b + "," + c + "," + d;
12                    System.out.println(solution);
13                 }
14              }
15           }
16        }
17     }
18  }
```

这是个不错的起点，现在怎么让它变得更快呢？

可以通过“走捷径”取得一些小胜利，比如当右侧已经太大时跳出，可以在 $a^3+b^3<c^3$ 时跳出。（令人吃惊的是，在 d 循环中检查等式并不能节省时间。在 d 太大时我们确实应该从最内层循环中退出。但对于所有比较小的 d 值，我们多运行几步。）

还可以通过去掉重复等式节省一点儿时间，考虑一下下面的解法，所有这些本质上都是相等的：

```
1   3³ + 60³  = 22³ + 59³
2   60³ + 3³  = 22³ + 59³
3   3³ + 60³  = 59³ + 22³
4   60³ + 3³  = 22³ + 59³
5   22³ + 59³ = 3³ + 60³
6   22³ + 59³ = 60³ + 3³
7   59³ + 22³ = 3³ + 60³
8   22³ + 59³ = 60³ + 3³
```

这些等式只需要输出一个。

可以强制让 $a \leqslant b$ 并且 $c \leqslant d$ 来去掉一些重复项。这样就可以避免输出 a 和 b 交换或 c 和 d 交换的等式。

然而还要杜绝等式左右两侧交换的情况，如果要求 $a < c$（对两组中一组而言这是真的），这种情况也可以避免了。

只要为 for 循环选择合适的开始条件，所有这些情况我们都能处理。如果让 b 从 a 开始，那么 b 永远都会大于等于 a。c 也可以同样处理，但让它从 $a+1$ 开始。（为什么要+1 呢？因为如果 $c=a$，则 $d=b$。$a^3+b^3=a^3+b^3$ 没什么意思。）

也可以得出 c 不会比 b 大的结论。因为 $a < b$ 并且 $c < d$。如果你想一下等式 $a^3 + b^3 = c^3 + d^3$。要让等式成立，不可能左边的两个值都比右边的两个值小。

```
1   void cubesBetter(int max) {
2      for (int a = 0;a < max;a++) {
3         int acubed = a * a * a;
4         for (int b = a; b < max; b++) {
5            int bcubed = b * b * b;
6            for (int c = a + 1; c < b; c++) {
7               intccubed = c * c * c;
8               if (acubed + bcubed < ccubed) break;
9               for ( int d = c;d < max;d++) {
10                 int dcubed = d * d * d;
11                 if (acubed + bcubed == ccubed + dcubed) {
12                    String sol = a + “,” + b + “,” + c + “,” + d;
13                    System.out.println(sol);
14                 }
15              }
16           }
17        }
18     }
19  }
```

这能有些改善，但时长还是 $O(N^4)$，还可以做得更好。

来看一下给出的等式：$a^3 + b^3 = c^3 + d^3$。一旦我们确定了 a、b、c 的值，则 d 的值也可以确定了。唯一的问题是是否有这样一个整型值存在。所以可以不用遍历所有可能的 d 值，只要检查 d 的结果值是不是整型。

```
1   void cubes(int max) {
2      for (int a=0;a < max;a++) {
3         int acubed = a * a * a;
4         for (int b = a; b < max; b++) {
5            int bcubed = b * b * b;
6            for (int c = a + 1; c < b; c++) {
7               int ccubed = c * c * c;
8               if (acubed + bcubed < ccubed)  break;
9
10              /* 计算(a3 + b3 - c3)的立方根，
11               * 并检查结果是否为整数。*/
12              int d = (int) Math.round(
13                 Math.pow((acubed + bcubed - ccubed), 1.0 / 3.0));
14
15              if ( d >= c && acubed + bcubed == ccubed + d * d * d) {
16                 String solution = a + “,” + b + “,” + c + “,” + d;
17                 System.out.println(solution);
18              }
19           }
20        }
21     }
22  }
```

这个是 $O(N^3)$。比原来的好了，但还不是最优解。

想想我们的算法做了什么，现在的方式相当于：

```
for each pair (a, b) where a < 1000 and b < 1000:
    compute cubeAB = a³ + b³
    find pairs (c, d) that sum to cubeAB
```

对于任何一组给定的数对，要遍历所有可能的另一组数对，检查它们是否相等。

实际上，可以将这些数对按和值分组，这样可以只是遍历一次数对。

在遍历这些数对时，可以创建和值→数对 (p,q)（`sum -> pair(p, q)`）的映射，然后输出每个和值内的所有数对的组合。也就是说，如果我们找到数对 $_1$、数对 $_2$、数对 $_3$、数对 $_4$，和值都是 x，我们可以输出(数对 $_1$, 数对 $_2$)、(数对 $_1$, 数对 $_3$)、(数对 $_1$, 数对 $_4$)、(数对 $_2$, 数对 $_3$)、(数对 $_2$, 数对 $_4$)、(数对 $_3$, 数对 $_4$)。

和值→数对 (p,q) 的映射看起来应该和这个差不多：

```
260245440 = 82³ + 638³
          = 144³  + 636³

958595904 = 22³  + 986³
          = 180³  + 984³
          = 692³  + 856³

8587000   = 46³  + 204³
          = 120³  + 190³

95880024  = 102³  + 456³
          = 228³  + 438³
...
```

可以用散列表实现这个，为简单起见，用一个字符串表示数对(a,b)。

```
1   void cubes(int max) {
2      /* 计算数对的和值。*/
3      Hashtable<Integer, ArrayList<String>> map = computeSums(max);
4      /*输出数对的组合。*/
5      printSolutions(map);
6   }
7
8   /* 创建从每个可能的和值到得出这个和值的数对的映射。 */
9   Hashtable<Integer, ArrayList<String>> computeSums(int max) {
10     Hashtable<Integer, ArrayList<String>> sums =
11        new Hashtable<Integer, ArrayList<String>>();
12     for (int a = 0;a < max;a++) {
13        for (int b = a; b < max; b++) {
14           int sum = a * a * a + b * b * b;
15           String solution = a + “,” + b;
16           /* 将 和值 -> 数对 添加到散列表中。*/
```

```
17            if (!sums.containsKey(sum)) {
18               sums.put(sum, new ArrayList<String>());
19            }
20            ArrayList<String> solutions = sums.get(sum);
21            solutions.add(solution);
22         }
23      }
24      return sums;
25   }
26
27   /* 输出每个和值的所有数对。*/
28   void printSolutions(Hashtable<Integer, ArrayList<String>> map) {
29      for (int sum : map.keySet()) {
30         ArrayList<String> solves = map.get(sum);
31         printSolutionsForSum(solves);
32      }
33   }
34
35   void printSolutionsForSum(ArrayList<String> solutions) {
36      for (int i = 0; i < solutions.size(); i++) {
37         for (int j = i + 1; j < solutions.size(); j++) {
38            String sol = solutions.get(i) + “,” + solutions.get(j);
39            System.out.println(sol);
40         }
41      }
42   }
```

这种解法的时长是 $O(N^2)$，其中 N 是 a、b、c 和 d 的最大值。

16.20　给出一个字符串，输出那个字符串的所有排列组合（假定这个字符串中没有重复的字符）。

227 页

这是一个典型的递归问题。

我们自底向下做一个例子来解决这个问题：

```
a -> a
ab -> ab, ba
abc -> abc, acb, bac, bca, cab, cba
```

怎么根据这些答案中的某个或全部做出 `abcd` 的所有排列组合呢？

主要区别是出现了 `d`，如果有 `abc` 的所有组合，就可以把 `d`“拼接”到这些字符串中（以所有可能的方式）。

```
abc -> abc, acb, bac, bca, cab, cba
       splice(abc, d) -> dabc adbc abdc abcd
```

```
splice(acb, d) -> dacb adcb acdb acbd
splice(bac, d) -> dbac bdac badc bacd
splice(bca, d) -> dbca bdca bcda bcad
splice(cab, d) -> dcab cdab cadb cabd
splice(cba, d) -> dcba cdba cbda cbad
```

下面的代码实现了这种解法：

```
1  ArrayList<String> permutations(String word) {
2     /* 基本情况：单词为空字符串。*/
3     if (word.length() == 0) {
4        ArrayList<String> list = new ArrayList<String>();
5        list.add(word);
6        return list;
7     }
8
9     /*去掉最后一个字符，得到剩余部分的排列组合。 */
10    String lastChar = word.substring(word.length() - 1);
11    String remainder = word.substring(0, word.length() - 1);
12    ArrayList<String> list = permutations(remainder);
13    ArrayList<String> result = new ArrayList<String>();
14
15    /* 遍历子字符串的所有排列组合，将 lastChar 插入其中。*/
16    for (String partial : list) {
17       /*将 lastChar 插入所有可能的位置。*/
18       for (int i = 0; i < partial.length(); i++) {
19          String left = partial.substring(0, i);
20          String right = partial.substring(i);
21          String spliced = left + lastChar + right;
22          result.add(spliced);
23       }
24       result.add(partial + lastChar); // 也插到最后
25    }
26    return result;
27 }
```

这个算法的时长是 $O(N!)$（其中 N 是字符串中的字符数），因为有 $N!$种排列组合。

不能优化这个算法，但还有另外一种方式。对大多数人来说不太直观。

对于字符串 abcd，可以像下面这样从子字符串的结果中做出来：

```
perms(abcd) =     {a + perms(bcd)}
                + {b + perms(acd)}
                + {c + perms(abd)}
                + {d + perms(abc)}
```

也就是说把每个字符串去掉并排列剩余的部分，然后再把被去掉的字符串放在每个组合结果的最前面。

可以不用在“子组合”前面添加每个字符，让子组合处理这件事情。permutations 函数可以接受一个前缀字符串，表示当前需要追加的字符，并对剩余部分进行排列。

```
1  void permutations(String word,String prefix) {
2     /*我们的前缀完全做好了，输出它。*/
3     if (word.length() == 0) {
4        System.out.println(prefix);
5     }
6
7     /* 去掉每个字符。排列剩余部分，传递前缀。 */
8     for (int i = 0; i < word.length(); i++) {
9        char c = word.charAt(i);
10       String left = word.substring(0, i);
11       String right = word.substring(i + 1);
12       permutations(left + right, c + prefix);
13    }
14 }
```

跟前面那种解法一样，这个也是 $O(N!)$。

16.21 在一群人中，所有人互相都认识，但其中也有不认识其他人的那种人，这种人被称为“名人”。给你一个函数 knows(a, b)，可以告诉你 a 是否认识 b。设计一个算法找出名人（如果有的话）。

简单起见，假定每个人都有一个标签，范围从 0 到 N-1。你需要实现一个函数 int findCelebrity(int N)。注意看：

(1) 最多只能有一个名人（基于名人的定义）；

(2) 给出的函数是找到谁认识谁的唯一办法。

227 页

先用一个简单的蛮力解法，遍历所有可能的人选，检查这个人是否是名人。一旦找到符合名人标准的人，就返回这个人。

```
1  int findCelebrity(int people) {
2     for (int i=0;i < people;i++) {
3        if (isCelebrity(people, i)) {
4           return i;
5        }
6     }
7     return -1;
8  }
9
10 boolean isCelebrity(int people, int candidate) {
11    for (int i = 0;i < people; i++) {
12       if (i != candidate) {
13          if (knows(candidate, i) || !knows(i, candidate)) {
14             return false;
15          }
16       }
17    }
18    return true;
19 }
```

因为可能要对每对人士调用 knows(a, b)，所以这种解法的时长是 $O(N^2)$。

看看还能不能做到更快。

想一想对两个人 x 和 y 调用 knows 的情况。knows(x, y)的结果或者是 true，或者是 false。从这些结果中能得出什么结论?

- 假如 knows(x, y) = true。说明 x 认识 y。说明 x 不是名人。名人谁也不认识。
- 假如 knows(x, y) = false。说明 x 不认识 y。则 y 不是名人，因为所有人都认识名人。

因此就可以得出一个很有趣线索：对于两个可能是名人的人，总是可以排除其中一个人是名人的可能性。

注意：现在是你停下来自己想出这种解法剩余部分的好机会。

如果总能排除一个人，那经过 N-1 次的 knows 调用，应该能将长度为 N 的人员清单缩减到只有一个人。那时候，可以对该候选人和其他所有人调用 knows，以检查那个候选人是不是真的名人（因为可能没有名人）。

这可以通过两步完成：

(1) 找到候选人；
(2) 验证那个候选人是不是名人。

在开始编码之前，先想想如何实现第一步：找到候选人。

有一个人员列表，当有人被排除时把人移除。在一个列表中偏移元素是个费事的操作，坦率地说，没必要做那么多工作。

想象一下我们对 knows 的调用，可以把这个算法看成把 0 从候选人中剔除出去。在调用 knows(0, 1)，或者排除 0，或者排除 1；如果排除 0，则候选人变成 1。然后继续调用 knows(候选人, 2)。

第二步还用我们之前实现的那个 isCelebrity 方法：

```
1  int findCelebrity(int people) {
2     int candidate = findCandidate(people);
3     if (isCelebrity(people, candidate)) {
4        return candidate;
5     }
6     return -1;
7  }
8
9  int findCandidate(int people) {
```

```
10     int candidate = 0;
11     for (int i = 0;i < people;i++) {
12        /*如果候选人被排除了，转向 i。*/
13        if (knows(candidate, i)) {
14           candidate = i;
15        }
16     }
17     return candidate;
18  }
19
20  boolean isCelebrity(int people, int candidate) {
21     for (inti = 0;i < people; i++) {
22        if (i != candidate) {
23           if (knows(candidate, i) || !knows(i, candidate)) {
24              return false;
25           }
26        }
27     }
28     return true;
29  }
```

这个算法的时长是 $O(N)$。

16.22 你有一个 $N \times N$ 的字母表，一个有效单词的列表（可以是你希望的任何格式）。你可以从任何一个字符开始，然后上下左右移动来形成一个单词。单词不必在一条直线上（比如下面这个矩阵中形成的单词 PACKING）。字母不能在单词中重复使用，所以 GOING（在下面的字母表中）不成立，因为它重用了 G。设计一个算法，并写代码输出所有有效的单词。

```
L I G O
E P N I
N A C K
S M A R
```

227 页

逐步考虑这个算法。我们需要找到以每个字母开头的所有单词，当然可以遍历字母表中的每个字母，开始搜索以那个字母开头的单词。

但如何找到以某个特定字母开头的所有单词呢，比如 P？

可以从 P 开始向上下左右移动，这说明可以将矩阵中所有从 P 开始的单词看成下面这样：

```
所有从 P 开始的单词 =
         所有从 PI 开始的单词
    +    所有从 PE 开始的单词
    +    所有从 PA 开始的单词
    +    所有从 PN 开始的单词
```

这很自然应该用递归算法，要在每个方向（前、后、左、右）递归，随着行进构建单词。只要有一个完整的单词，就输出它然后继续递归。

其中一个比较需要技巧的地方是如何防止在相同的单词中重用一个字母。这有几种解法，但所有解法都在遍历时用相同的方式将字符标记为“已用”。之后再去掉标记，重新使用。

用一个布尔型数组来解决这个问题。在遍历到一个单元的隔壁单元之前，先将这个单元标记为已被占用。做好之后，再将它标记为可用。

还可以在递归中及早通过捷径执行优化。比如在形成字符串 PNCKR 时，词典中肯定没有以它开头的单词，那么为什么还要沿着这条路径继续递归呢？

如果将词典做成一棵字典树，就可以用一个函数来告诉我们某个字符串是否是词典中某个单词的子字符串。字典树是一种特殊的树，一般用来存储单词。使用字典树时，调用 isPrefix 方法的效率非常高。

在这个算法中，可以用 isPrefix 方法来提前终止非有效字符串上的递归：

```
1   /* 通过寻找以字母表上每个字母开头的单词寻找字母表上的所有单词。*/
2   void boggle(char[][] board) {
3      boolean[][] marked =
4         new boolean[board.length][board[0].length];
5      for (int i = 0; i < board.length; i++) {
6         for (int j = 0; j < board[0].length; j++) {
7            boggle(board, i, j, “”, marked);
8         }
9      }
10  }
11
12  /* 找到所有以前缀开头的单词并使用行、列上的字母。 */
13  void boggle(char[][] board, int row, int col, String prefix,
14              boolean[][] marked) {
15     /* 检查字母表上的字母当前是否已经用上了。*/
16     if (!inBounds(board, row, col) || marked[row][col]) {
17        return;
18     }
19
20     /* 将字母追加到当前的单词上。*/
21     prefix = prefix + board[row][col];
22
23     /*如果没有以这个前缀开头的单词，返回。*/
24     if (!isValidPrefix(prefix)) {
25        return;
26     }
27
28     /*找到一个单词。输出它。*/
29     if (isValidWord(prefix)) {
30        System.out.println(prefix);
31     }
```

```
32
33     /* 将字母标记为已占用。*/
34     marked[row][col] = true;
35
36     /* 遍历它的每个邻居。*/
37     boggle(board, row - 1, col, prefix, marked); // 向上
38     boggle(board, row, col + 1, prefix, marked); // 向右
39     boggle(board, row + 1, col, prefix, marked); // 向下
40     boggle(board, row, col - 1, prefix, marked); // 向左
41
42     /* 遍历完了它的邻居，现在返回到它的父单元。将这个单元标记为可用的。*/
43     marked[row][col] = false;
44 }
45
46 /* 检查行、列是否在边界内。*/
47 boolean inBounds(char[][] board, int row, int col) {
48   if (row < 0 || col < 0 ||
49       row >= board.length || col >= board[row].length) {
50     return false;
51   }
52   return true;
53 }
```

描述这个算法的运行时长有点困难，因为这真的要看字母表和英语的特点。如果很多路径都是有效的（即形成有效前缀的单词），那它将会比很多路径无效时慢很多。

如果不做 `isPrefix` 检查，要遍历 N^2 个字母，对于每个字母，都要向四个方向移动一次，然后是三次。一条路径的长度最长可能是 N^2（字母数），所以从某个给定字母开始的可能路径是 $O(4 \times 3^{(N\wedge2)})$。这样时间就是 $O(N^2 \times 4 \times 3^{(N\wedge2)})$，去掉常量降为 $O(N^2 \times 3^{(N\wedge2)})$。

实际上，考虑到字典树和英语中单词的模式，真实结果要快得多。

16.23 给定一个整型数组（既有正值又有负值），找到和值最大的连续序列（只返回和值）。

比如：

```
输入：2, -8, 3, -2, 4, -10
输出：5 (即 {3, -2, 4})
```

227 页

先从蛮力法开始，然后看看怎么优化它。

方法 1：蛮力法

我们可以遍历所有可能的子序列，将他们的和值跟最大的和值比较。最后返回我们见到的最大值。

```
1  int getMaxSum(int[] a) {
2     int maxSum = 0;
3     for (int left = 0; left < a.length; left++) {
4        for (int right = left + 1; right < a.length; right++) {
5           int sum = 0;
6           /* 加上所有中间的值 */
7           for (int i = left; i <= right; i++) {
8              sum += a[i];
9           }
10          if (sum > maxSum) {
11             maxSum = sum;
12          }
13       }
14    }
15    return maxSum;
16 }
```

这是 $O(N^3)$。我们可以做得更好！

方法 2：蛮力法（优化）

因为每个子序列都有唯一的起点和终点，我们知道一个数组大概有 $O(N^2)$个子序列。而之前那种算法所需的时间是 $O(N^3)$，这说明它是有优化的可能的。

看一下最里层的 `for` 循环（`i` 循环）是做什么的。它只是计算 `left` 和 `right` 中间的所有元素的和值，而我们刚刚算过 `left` 和 `right-1` 中间所有元素的和值（在之前的 `right` 循环中）。

不用每次都重新计算和值，只需要维护一个变化的和值。

当 `right` 进入下一次循环时，只需把 `a[right]`加到变化的和值上：

```
1  int getMaxSumBF(int[] a) {
2     int maxSum = 0;
3     for ( int left = 0; left < a.length; left++) {
4       int runningSum = 0;
5       for (int right = left; right < a.length; right++) {
6         runningSum += a[right];
7         if (runningSum > maxSum) {
8           maxSum = runningSum;
9         }
10      }
11    }
12    return maxSum;
13 }
```

现在时间降到了 $O(N^2)$。好了一些，但这个基本上还是使用蛮力的解法。

方法 3：优化法

我们深入探查一下最后这种解法究竟做了什么。

```
2, -4, 4, -3, 2, 5, -1, -4, -5, -2, -1, 2
```

我们遍历了所有可能的子序列。那包括，比如，包含前两个值的子序列（2 和-4）。为什么要一个以{2, -4}开始的子序列？它们的和值是-2，也就是说它们只会让子序列的和值变小。

有时我们确实想要包含负值的子序列，但那只是在负值可以连接两边更大的值时。

这样可以得出一条结论：当一个子序列为负时，就不应该包含它。

改一下代码，在 runningSum 为负时及早退出，这样就可以尝试下一个 left 值了。

```
1   int getMaxSum(int[] a) {
2      int maxSum = 0;
3      for (int left = 0; left < a.length; left++) {
4         int runningSum = 0;
5         for (int right = left; right < a.length; right++) {
6            runningSum += a[right];
7            if (runningSum > maxSum) {
8               maxSum = runningSum;
9            } else if (runningSum < 0) {
10              break;
11           }
12        }
13        return maxSum;
14     }
15  }
```

经过这次修改，现在一经过{2, -4}我们就跳出了。left 会跳过-4，然后指向 4。

继续移动 right，直到 runningSum 变成负值。当 runningSum 比 maxSum 大时，更新 maxSum。

runningSum 什么时候会变成负值？我们来演练一下：

```
left:          4
right:         4 | -3 | 2 | 5 | -1 | -4 | -5 | ...
runningSum:    4 |  1 | 3 | 8 |  7 |  3 | -2 |
maxSum:                     8
```

当 right 指向-5 时我们跳出来了，现在我们肯定找到了从 left 开始的最大子序列。

注意一下，到那一点时，left 和任何一点 x 之间的值都大于或等于 0。换句话说：

```
sum(array[left], array[left + 1], ..., array[x-1]) > 0
```

想象有一个从 x 开始的子序列继续沿着数组向后。如果 sum(array[left], array[left + 1], ..., array[x-1]) > 0，则所有从 x 开始的子序列都不如从 left 开始的大。因此从 x 开始不是更优的选择。

所以，我们还没找到从 left 开始的最大子序列，而是在 left 和 right 直接任意位置开始的最大子序列。

现在应该将 `left` 移到 `right + 1`。

这样就得出了一个新的算法。

(1) 让 `left` 和 `right` 从最左侧开始。
(2) 移动 `right`，直到 `runningSum` 变成负值。
  » 沿途追踪 `runningSum` 和 `maxSum`。
(3) 当 `runningSum` 变成负值时，将 `left` 移到 `right + 1`，并重置 `runningSum`。

```
1   int getMaxSum(int[] a) {
2      int maxSum = 0;
3      int runningSum = 0;
4      int left = 0;
5      for (int right = 0; right < a.length; right++) {
6         runningSum += a[right];
7         if (runningSum < 0) {
8            left = right + 1;
9            runningSum = 0;
10        }
11        if (maxSum < runningSum) {
12           maxSum = runningSum;
13        }
14     }
15     return maxSum;
16  }
```

如果你仔细看，会发现虽然设定了 `left`，但从没真正用过，因此可以从代码中把它去掉。

```
1   int getMaxSum(int[] a) {
2      int maxSum = 0;
3      int runningSum = 0;
4      for (int right = 0; right < a.length; right++) {
5         runningSum += a[right];
6         if (maxSum < runningSum) {
7            maxSum = runningSum;
8         } else if( runningSum < 0) {
9            runningSum = 0;
10        }
11     }
12     return maxSum;
13  }
```

到此我们的优化完成了。这段代码的运行时间是 $O(N)$，只要遍历一次数组就能计算出和值最大的序列。不可能做到更好了。

附　录

附录包括大量能助你作面试准备工作的资源。如果需要更多资源，或意欲与同行产品经理交流讨论问题，请参阅CrackingThePMInterview.com。

A. Ian McAllister：前 1%的产品经理与前 10%的产品经理

麦卡利斯特（Ian McAllister）是亚马逊慈善项目 AmazonSmile 的创始人和领导者。他领导产品管理团队、软件开发团队、用户体验设计团队，还负责部分业务开发工作。在此之前，他主管亚马逊的全球礼品业务，而且是微软的项目经理。

是什么使得前 1%的顶尖产品经理与前 10%的产品经理有所区别？

以下枚举的这些特质，前 10%的产品经理只擅长其中若干，而前 1%的顶尖产品经理则擅长其中大多数或全部。

- **大局思想**　位居前 1% 的产品经理的思维不会被当前市场环境或可用资源所约束，他们会描绘颠覆性的大机遇，并制定具体计划以从中获利。
- **沟通能力**　位居前 1% 的产品经理的提案是不可能被反驳或忽略的。他们会在恰当的时机恰当地使用数据，也会利用决策者的偏好、信仰或软肋来说服他们，让他们乖乖带着人员、经费或其他资源参与进来，并且不再阻挠。
- **简约化**　位居前 1% 的产品经理知道如何投入 20%的努力，而从任何产品功能或项目中谋取 80%的价值。他们乐此不疲，不断发布更多产品，并为产品或业务带来盈利。
- **分清事物主次**　位居前 1% 的产品经理知道如何对项目主次优先级进行排序。他们能在速效方案和平台投资之间找到平衡，也能在主动出击项目与被动防御项目中取得平衡。主动出击项目是指那些能增长业务的项目，而被动防御项目是指那些能够保护现有业务、移除业务累赘（运营、偿还“技术债务”、修复 bug，等等）的项目。

- **预估与测量**　位居前 1% 的产品经理能够预估一个项目的大致收益，还能够利用经验和具有可比性的参照标准对此进行预估。他们还能在项目发布之后对收益进行测量，并将所学到的经验应用到未来的项目优先级和收益预估中。
- **执行能力**　位居前 1% 的产品经理什么事都肯努力去做。只要是上线必需的，他们都会尝试。因为他们明白的职能范围是没有明显界限的。只要有必要，他们就会招募人员，制作按钮，开发业务，升级产品，甚至与内部法律顾问角力……
- **理解技术上的权衡取舍**　位居前 1% 的产品经理不一定拥有计算机科学学位，但他们需要对待开发产品的技术复杂性有大致的理解，而不需开发部门投入成本来帮他理解这些问题。他们应该与开发部门合作，来作出最佳的技术取舍（即妥协）。
- **理解优秀设计方案**　位居前 1% 的产品经理不必非得是设计师，但应该能够鉴赏优秀的设计方案，并能从良好的设计方案中辨别出优秀卓越的设计方案。他们也应该能够向设计师同事清楚地表达方案之间的区别，或至少清晰地指出从“良好”到“优秀”的方向。
- **撰写可用文档**　位居前 1% 的产品经理应当能够撰写简洁且能奏效的文档。他们应当知道，每多写一个词语，前文所述的价值就被冲淡了一点。他们应当花费时间和精力，为重要文档（按钮标签、导航、行动按钮等）找到最佳的表述方法，而不仅是一般合格的表述。

我似乎从未遇到过一个位居前 1%的顶尖产品经理，当然在雇佣之前不会遇到这样的人。你不必一开始就去寻找这 1%的顶尖产品经理，可以试着去雇佣一个位居前 10%而且有发展潜力的候选人。

原文发表于：http://www.quora.com/Product-Management/What-distinguishes-the-Top-1-of-Product-Managers-from-the-Top-10/answer/Ian-McAllister。

B. Adam Nash：如何成为一名优秀的产品经理

亚当·纳什（Adam Nash）是财富前沿公司（Wealthfront）公司的首席运营官。在加入 Wealthfront 之前，他是格雷洛克合伙企业（Greylock Partners）的驻企高管，为该企业旗下消费技术公司的领导团队提供顾问咨询并评估风投机会。在此之前，他曾在 LinkedIn 担任产品管理副总裁。

认识我的同行都知道我热爱产品管理。我坚信，只要方法得当，一个强劲的产品领导者就像力量倍增器一样，能够帮助一个由优秀技术人员和设计师组成的跨职能团队将工作做到最好。

不幸的是，产品经理这个职位的职责说明要么过于模糊（“你要负责某产品”），要么过于具体（“你要为产品编写产品说明书”）。事实证明，二者都无法有效帮助人们成为优秀的产品经理。

我花了不少时间试图找到一个方法，来说明产品经理的价值，即能清晰地向跨职能搭档传达他们应从产品经理处得到（或要求）什么，同时也能向新的产品经理传达他们对工作的实际期望。过了几年，我将这种信息传达削减为 3 项职责：产品策略、优先级及执行力。

职责 1：产品策略

顶尖的商业学校都会教授关于战略的完整课程。然而，我不觉得你会在那些课程中听到有人以这种方式来讨论产品策略。很简单，产品经理的职责就是清晰表达这两件简单的事情。

- 我们在玩什么游戏？
- 我们如何计分？

一旦你将这两件事做对了，一大堆在工程、运营、质量、设计和市场上的卓越天才豁然间就开始往同一方向奔跑。要是没做对这两件事，什么优先级、什么执行力管理都救不了你。设计优秀的软件需要各种各样的人才，而其中关键性的创意可以出自任何地方。清楚描述你们正在玩的游戏和衡量成功的标准，可让团队在不依赖产品经理的情况下，将不同的想法分类整理，并决定哪些值得付诸行动。

清晰定义你们在玩的游戏，包括你对产品的愿景、为用户提供的价值以及你区别于竞争者的优势。然而更重要的是，这样做能够清楚描绘出具体的方法，让你的团队在市场上所向披靡。我们不妨假设你选定了适当的衡量标准，那么团队中的每位成员都应当对成功有一个清晰的概念。

对于那些哪怕只在岗位上工作了 2 个星期的产品经理，你可以问他们这两个问题，得到的答案应当不仅清晰，还令人信服。

总结：齐心合力、积极主动、创意思维，以及有影响力的产品。

职责 2：分清优先级

一旦团队清楚了他们在玩的游戏以及如何计分，要对工作划分优先级就容易多了。这是产品经理的第二项职责——确保团队制定产品战略和产品衡量标准始终贯穿整个项目或产品功能的各个阶段。

在人才济济的公司里，好的创意想法比比皆是。其实，并不是规模越大，制定优先级的情况就越好，因为加入公司的人越多，想法也越多。结果，野蛮划分优先级就成了常事。

问题不在于按业务需要列出一张最佳创意想法的优先级清单，而是找到团队接下来要执行的三件最重要的事情。

给工作分阶段，是任何企业家奋斗史的重要部分。绝大多数产品和公司的失败并不是因为缺乏好创意，而是因为没有分清事情的主次优先级——哪些最为关键，应最先执行；哪些可以稍后处理。

我个人认为，长期看来，线性优先级划分是无效的。我曾在一张独立的帖子中写到“3 个水桶”的产品优先级划分，它阐明了我所主张的优先级划分流程。

对于那些哪怕只在岗位上工作了 2 个星期的产品经理，你可以向他们索取他的团队正在进行的项目的优先级列表，理想状态下，他应当对整个优先级的划分有充分合理的理由。

职责 3：执行力

实际上，产品经理能做很多事情。

产品最后要由产品经理来宣布推出，这就意味着产品经理要在生产过程中覆盖所有需要覆盖的缺口。有时他们要创作产品内容，有时候要修补设计漏洞，有时候他们要管质量问题，有时候他们还要处理公关问题。总之，只要是在人类的能力范围内，他们会做任何能让产品成功所需要做的事。

然而，有一部分的执行工作对团队来说极其重要。没有了这些，团队的执行工作就会变得极其低效。

- **产品规格说明书**　说明书应具产品所必要的详细程度，以确保团队了解所要创建的产品；
- **边缘情况决策**　项目过程中，经常会不经意冒出一些复杂的边缘情况。一般来说，产品经理就是要整装待发，时刻准备快速诊断出那些可能会影响产品其他部分的问题；
- **项目管理**　对任何功能点来说，产品经理总要面临“时间/收益”的权衡取舍。在产品周期中，很多这样的产品需求最终被砍掉。因此，产品经理必须走在潜在的问题前面，抢先确保产品最终能在正确的时间点推出并赢得市场；
- **分析**　最后，团队要大力依赖产品经理来分析各种数字，并得出详细结论，知道哪些功能点对产品功能目标起了至关紧要的作用。此外，他们还期望产品经理深入分析现有的功能点以及竞争者功能点（如果有的话）。

让事情得到解决

最后，优秀的产品经理能让事情顺利终结。如果一位优秀的产品经理加入你的团队，你总能可靠且准确地将他与普通产品经理区分开来，因为他一来，很多问题就开始被解决了：bug 的修复和功能的优化开始进行了，数据的清晰分析出来了，项目重新被划分了优先级。用不了多久，衡量标准的关键数字开始往上飙升，趋向正常了。

去吧，去做一个优秀的产品领导者吧。

原文发表于：http://blog.adamnash.com/2011/12/16/be-agreat-product-leader/。

C. Sachin Rekhi：一份卓越的产品发展蓝图需要什么投入？

Sachin Rekhi 是一位具有产品经理背景的多次创业者。他创立了 Connected（后被 LinkedIn 收购）、Feedera（后被 LinkedIn 收购）和 Anywhere.fm（后被 imeem 收购）。现在，他是 LinkedIn 一名集团产品经理，之前也曾在微软任项目经理。

经常有人问我该如何为一款即将发布的产品设计产品发展蓝图。以下，我就会将自己近期带领团队为一款新产品设计蓝图的经历分享给大家。

现有的使用衡量标准分析

当你想方设法要改进现有产品时，最好的方式是从现有产品的使用模式的深入分析开始。深入理解最常被使用的产品功能，可以让你知道未来最为合理的投资方向；不常用的功能也能提醒你，哪些功能需要重新设计或直接移除。深入流程分析还可以帮助你理解，在目前的用户体验下，可以对产品进行哪些优化，从而优化用户体验。

进行用户访谈，以了解受众的痛处

伟大的产品能为重要问题提供解决方案，因此，关键是要确保你正在解决的正是目标受众最紧迫的问题。我发现，要真正了解受众的痛处，最好的方法就是进行用户访谈，设法理解受众的使用动机、日常工作流、现有工具、目前的烦恼，等等。此时应专注于问题空间，而非解决方案空间，这样才能真正明白，要解决的是哪些问题。

顾客反馈和支持请求的聚合分析

用户时常会联系上产品的发布者，为他们提出功能建议、支撑请求、投诉抱怨等反馈意见。花费时间来总结这些反馈意见，并以此了解用户群的偏好和窥探未来投资的方向，这些都是很有价值的。

深入探究市场竞争

了解市场上其他玩家的情况，看看他们是如何做得风声水起，或者为何会鸦雀无声 。这是产品发展蓝图中产品创意的另一个重要来源。亲手试用各种产品、逛产品用户论坛、阅读产品和行业评论，可以帮你发现市场竞争上最有意思的东西。对比竞争对手的情况来玩自己的游戏确实很重要，这也是产品发展蓝图的一项重要的投入。

内部创意商业化

很多情况下，你的产品只是公司一系列产品中的一个组成部分而已。每个产品创意都趋于为其本身量身定制，并将对受众与产品领域而言最为重要的东西推出市场。尽管如此，时常还是会出现这样的情况：你所推出的“新”创意产品类似于公司产品系列中其他产品。因此我认为，了解公司其他产品的最新发布情况是很有用的，这能让我们知道在自己的产品领域中，有哪些东西是可以借鉴利用的。

进行受众调查，以了解功能优先级

一旦你绞尽脑汁想出了一系列潜在的产品功能，你可以试着对现有或潜在的用户作一项调查，以此来帮助你划分功能优先级，这常常是很有用的。借助于联合分析，你可以得出每个产品功能点的相对重要性，以便考量后续资源的分配情况。

这些工作都有利于你筛选出下次的关键投入点，并帮助你制定产品发展蓝图。然而，要记住，开发出恰当的产品发展蓝图既是一门艺术，也是一门科学。这些投入可以帮你描绘产品发展蓝图，反过来说，一张优秀的产品发展蓝图就是这些投入的最终结果。

原文发表于：http://www.sachinrekhi.com/blog/2013/09/23/the-inputs-to-a-great-product-roadmap。

D. Ken Norton：如何选聘产品经理

肯・诺顿（Ken Norton）是谷歌风投（Google Ventures）的一位合伙人。在那之前，他是一名谷歌的集团产品经理。他在 2006 年通过 JotSpot 招聘计划以产品副总裁的身份加入谷歌。此前，他是雅虎产品管理高级总监。

很久之前，我在一家初创公司做过招聘。初创公司的人员招聘显然和大公司不同。在雅虎搜索，我们好像总是不断在招聘，平均一周我得参加 5 到 8 场面试。简历、面试、录取通知永无休止，一个接一个来来去去。后来，我就不经常做招聘工作了，只负责招聘一小部分产品经理。在大公司的招聘中，你会首先注意到一点，那就是他们对专业化的要求；在初创公司中，员工基本上什么事都要做一点儿，所以他们需要的是通才。更重要的是，未来难以预测，所以你需要一些适应能力强的人。你最初可能是要招聘某个人来做特定的工作，但几个月后这项工作可能会有所变化。大公司就不会出现这种情况。他们在招聘时已经很清楚，这个岗位需要的是什么样的人，而且这个岗位的工作职责改变的可能性很低。在雅虎雇员当中，很多人应该都不适合初创公司。我记得，很多招聘者在面试后会有类似这样的对话：“嗯……其实我太确定他们是不是最适当的人选了，但是看起来他们好像特别适合这个岗位。好吧，那就聘用他们吧。”这种情况对大公司来说可能没什么问题，但初创公司要是这么干，可就太糟了。

我最初是一名工程师，很快我便升为工程管理层。在互联网泡沫时期，我大概招聘了一百来名工程师。通过不断地在错误中学习，我在招聘这件事上也学到了不少。当我转作产品管理时，我能将招聘技术类人员的经验用来招聘产品经理，但我从中也得了不少全新的教训。上周一个朋友打电话来，说他需要招聘一名产品经理，希望我给点建议。我意识到，现在外头并没有一套完善的聘用产品经理的技巧（大体上也没有一套完善的产品管理技巧）。说得更确切些，不管你身处什么环境，初创公司也好，大公司也好，在招聘产品经理时应当对他有什么样的期望，现在都没有太多准则。所以，我觉得是时候要总结一下我学到的那些经验了。

记住，没人叫你表现自己

产品管理可能是唯一一个可有可无的工作——哪怕没有这个岗位，公司也能良好运转（至少可以运转好长一阵子）。没了工程师，什么产品也无法构建；没了销售人员，公司就卖不出任何产品；没了设计师，产品就惨不忍睹。但是在没有产品经理的世界里，大伙儿各司其职，各补其缺，照样工作。你得记住这一点：作为产品经理，你是可以被牺牲掉的。从长远来看，产品管理一职通常能在输赢之间起决定作用，这就看你的了。产品管理也要求你具备众多专业知识——工程、设计、营销、销售、业务开发，等等。产品管理这门学科到处都是怪胎和弃儿，这些人都在其他任何领域都显得格格不入。就拿我来说吧，我热爱工程上的技术挑战，但厌恶写代码；喜欢解决问题，但讨厌别人对我的比划指点。我想要成为作战略决策的一员，想要主导产品。市场营销要求我具备创造力，但我很清楚我不想远离技术太多。工程师尊重我，但他们知道我的心思在别处，而且他们普遍又觉得我太“市场化”了。像我这样的人，天生就更适合搞产品管理。

1. 聘用聪明人

那么，我在产品经理身上会看中什么特质呢？最重要的一点，天资聪颖。我会接受那些毫无经验但有些狡黠小聪明的产品经理，而不是那些有多年经验但天资平庸的产品经理。从根本上说，产品管理就是要站在你的立场考虑问题，总要抢先走在竞争者前面，并且还要能够设身处地，想同事和客户所想。我通常会向候选人发问一连串分析性问题，以考验他们的智力和解决问题的能力。我一般会一直发问，直到我确保他比我聪明为止。出于某些原因，我认识的很多人都不太愿意这样做，他们认为连续发问对候选人很无礼。但是我认为，一名合适的候选人会喜欢这种挑战。实际上，这就是他们的第一关考验——当我说“我想提出几个理论问题，可以吗”，他们应该会作何反应。最优秀的候选人通常的反应是非常兴奋，而那些超级聪明的候选人还会提出一些问题来回击面试官。

2. 强大的技术背景

我认识的一些管理者坚持要聘用拥有计算机科学学位的产品经理。我可没那么势利眼（可能是因为我本身接受的就是人文科学本科教育），不过我也会偏好于那些担任过技术角色的人。扎实的技术背景能让产品经理具备两项关键技能：与工程师沟通的能力，以及对驱动产品的技术细节的掌

控能力。当然，这取决于产品——比起交友网站前端页面的产品经理，一名初级开发者 API 项目的产品经理势必要懂更多的技术知识。但是，基本原则是普遍适用的：在向工程师传达产品需求，以及向没有技术背景的同事或客户传达复杂的产品细节时，拥有技术背景的产品经理能做得更好。即使如此，还是有一些问题需要你处处小心。尤其对于曾经是技术工程师的产品经理来说，需要清楚的关键一点是：你已经不再是原来的技术工程师。有些由工程师转型的产品经理仍会试图介入技术决策和实施细节，这势必会让产品管理乱了套。因此，我喜欢聘用那些在上一份工作中就已经转型为产品经理的技术人才。他们已经度过了艰难的适应阶段，通过证明人，我还可以知道他们适应得如何。我可不想用一些评估技术能力的面试问题来糊弄你。问哪些问题，得依据要考验的具体技能而定。而且，外头还有几百个网站为聘用工程师支招。我这儿倒是有一些不错的问题，可以用来评估一名技术型产品经理对他的角色的适应程度以及他与工程师的共事能力。

- 你为什么决定从技术工程转做产品管理？
- 拥有技术背景最大的优势是什么？
- 最大的劣势又是什么？
- 从技术工程转向产品管理的过程中，你最大的收获是什么？
- 有什么事情是你希望你在做工程师时就知道的？
- 如何赢得技术团队的尊重？

3. 蜘蛛侠般敏锐的产品直觉与创意

下面要说的这种能力是高度主观、难以评估却又举足轻重的。我坚信，有些人天生就有极好的产品直觉，他们就是知道怎么样才能成就一个好产品。他们并不总是对的，但他们的直觉常常能指向正确的方向。他们往往偏执于某种观点，有时会因此惹恼同事。我曾有幸与许多这样的人共事过，而且这是产品经理必备的基本品质。这种品质可以调校，但却学不来。产品管理要涉及到许多小决策，尤其在瞬息万变的环境中（如网站）。当然了，也有很多大的考量和战略。但就是这些小决策将优秀的产品经理和合格的产品经理区分开来。你要知道，他们之所以拥有“蜘蛛侠般敏锐的产品直觉”，就在于在他们提出的解决方法是团队中其他人想不到的，能在瞬间击中所有人的要害。要在面试中评估候选人的产品直觉是最具挑战性的，但这可以做到。我一贯的做法是确认候选人能否在一小时的面试中完成以下的任务。

- **能独立说出一些我对我负责的产品的担忧**　如果你是一个好产品经理，就会知道自己的产品有哪些事情需要操心。这可能是一些 UI 设计的不足、功能上的缺失，或是需要解决一些信息架构的缺陷。你很清楚，这些问题是需要修复的。而对于聪明且有强烈产品直觉的局外人来说，这里面肯定至少有一些问题是很明显的。我很希望在面试中能碰到这样一些人：他们的回答能让我微笑、点头，心领神会地说：“就是嘛！我们也快要被这个问题逼疯了！”
- **告诉我关于我产品的一些新想法**　可以是一项我未考虑过的改进，一个与对付竞争者的新想法，或是一个他们曾遭遇过的需要解决的问题。如果我能从候选人身上学到东西，

我就知道了两件事：(1) 他们敢于直言；(2) 他们可能比我聪明。于是，我想要的产品经理的两种特质，一下子就都满足了。

- **向我介绍一些新鲜有趣的东西**　拥有良好产品直觉的人往往比任何人都先注意到优秀的产品。如果我面试的是一名顶尖的候选人，在结束面谈时，我总会收获一些新鲜有创意的东西。

下面是判断产品直觉一些不错的问题。

- 谈谈你最近遇到的一个优秀产品吧。你为什么喜欢这款产品？（顺便说一下，有些候选人遇到这种问题时就回答说是我的产品，这种人挺让人难堪！比如在雅虎时，就有些候选人告诉我他们遇到过最酷的产品就是雅虎。）
- 是什么使某产品获得成功？（我通常会选一个流行的产品，比如 iPod 或 eBay，这些产品都在激烈的市场竞争中轻而易举地赢得消费者。）
- 我的产品有哪些地方是你不喜欢的？你会如何改进这个产品？
- 在今后的 1 年、2 年或 10 年里，我们将会遭遇什么样的问题？
- 你如何判断一款产品的设计是否合理？
- 你有过最牛的想法是什么？
- 你有过最糟的想法是什么？
- 你如何判断何时该走捷径，以便尽快发布产品？
- 你在 UI 设计方面得到过什么教训？
- 你如何决定什么要做，什么先不做？
- 你在产品上犯过最大的错误是什么？
- 你觉得产品管理哪些方面最无趣？为什么？
- 你认为自己是个有创意的人吗？

4. 赢得领导力

产品经理在团队中通常是领导者，但他们对其他成员一般并没有直接的职权。这就意味着他们要去赢得权威，并通过自己的影响力来领导团队。对于产品管理来说，领导能力和人际交往能力都非常关键。市面上大概有上千本书都是说领导力的，我就不赘述了——反正这些书大部分是垃圾。我发现，背景调查是评估领导能力的最有效方法，尤其是向那些与候选人共事过的同级同事（而不是候选人的上级领导）了解情况。不过，如果要当面问候选人，下面是我过去常常会问的几个问题。

- 达成共识是否总是好事？
- 管理与领导有什么区别？
- 你喜欢与什么样的人共事？
- 你觉得哪类人在工作中很难相处？

- 谈谈你以前的团队某次无法凝聚起来的情况。你认为其中的原因何在？你从中学到了什么？
- 如何使团队按照排好的时间表完成工作？
- 别人做什么事会使你对他失去信心？
- 你是否使用不同的管理方式来管理不同职能的人？如果是，如何管理？
- 如何拒绝别人的要求？
- 谁是产品发布的最终责任人？
- 你的团队是否曾使你失望，并且要求你来承担过错？
- 这些年来，你的容错度是如何变化的？
- 好消息和坏消息，你喜欢先听哪一个？
- 你的招聘方法是什么样的？

5. 多角度分析问题的能力

做一名产品经理，头上需要戴好几顶帽子。我常开玩笑说，产品经理的多数工作时间是在替那些并未共处一室的人说话的——用户、工程师、销售人员、主管、营销人员，等等。这就意味着你需要有能力去做其他人的工作，但又足够聪明，知道用不着真正去做。优秀的产品经理知道如何去整合不同的意见。他们会经常唱反调、抬杠。他们往往不满足于简单的答案。他们可能才刚说过某个需求在技术上不可行，下一秒他又会问该如何让销售人员明白这些需求。有一种明显的方法，可以评估候选人多角度考虑问题的能力：让更多人来面试这位候选人。我总是坚持，至少要由工程、设计、营销部门的代表人员来面试产品经理候选人。根据面试岗位的不同，面试官还可以增加：售前工程师、技术支持人员、开发者关系、业务开发人员、法务人员或者客户。往大了说，任何未来有可能与候选人共事的人都应该和他见一面。注意，我并不是说每个人都“必须”见一见候选人。每个职能中精心挑选一位代表就够了。而且，我也不是说，所有人都要赞许这名候选人才行。随着面试官的增加，一场面试很难达成一致的意见，所以要合理地考虑他们的反馈意见即可。但是话说回来，没有其他人能像销售人员那样判断一名产品经理对销售过程有多了解。我也强烈建议你给面试官明确的指示，如“我想要你看看，这名候选人能否充分理解你在渠道开发中遇到的问题，以及他能在这个领域里如何配合你的工作”。下面是我常用的一些具体问题（这些只是举例，你可以随意替换掉其中的职能名称）。

- 与销售人员共事，你学到了什么？
- 与用户互动的最好方法是什么？
- 市场营销是如何运作的？
- 如何判断产品的设计方向是否对头？
- 产品经理应当如何配合业务开发的工作？
- 关于向上管理，你学到了什么？
- 与高管人员共事的最好方法是什么？

6. 有代表作品的人

最后一种特质可能是最好评估的。除非这是一个非常初级的岗位，否则我一般会聘用一名曾出过产品的产品经理。出过产品，我是指要从头到尾，参与了从概念到推出的整个过程。除了自己出过好产品之外，没有什么更能体现一个人的产品交付能力。过去的成绩能映射出未来是否成功的可能性。这样做还有一个好处，就是它能在不可估摸的无形海洋中给出一些有形的东西让你来评估。向证明人核查信息时，我通常会确保与候选人先前所在的项目团队中重要的同事谈话，尤其是候选人的经理和与他接口的技术人员、销售人员或营销人员。（顺带说一句，这些规则是按照一定的原因排序的，但正如我第 1 条中提到的，相对于有多年经验而资质平庸的产品经理，我更愿意接受那些天资聪颖的产品经理，哪怕他们没有出过产品。）

后记：写下这篇文章时是 2005 年，当时我还供职于 JotSpot。谷歌在 2006 年收购了 JotSpot。从那时起，我就开始有机会接触一些出色的产品经理，还参加了 200 多场产品经理面试。在这几年里，我的观点肯定是有所改变的，但关于出色的产品经理应具备的特点的想法，倒是更加深刻了。我偶尔会有更新这篇文章的想法，但最终还是决定让它保持原样。

原文发表于：https://www.kennethnorton.com/essays/productmanager.html。

E. 亚马逊领导力准则

以下的领导准则转载自 Amazon.com[①]。如果你要申请亚马逊的工作，你绝对应该熟读它们。这些准则应该适用于其他公司，因为很多公司也都在寻找相同的人才。

不管你是大型团队的个人贡献者还是经理，你都是一名亚马逊式的领导者。这些就是我们的领导力准则，每一名亚马逊人都以这些准则为指导。

顾客至尚

领导者从客户入手，再反向推动工作。他们努力地工作以赢得并维系客户对他们的信任。虽然领导者会关注竞争对手，但是他们更关注客户。

主人翁精神

领导者是主人翁。他们会从长远考虑，不会为了短期业绩而牺牲长期价值。他们不仅仅代表

① 译文引自亚马逊中国，未作改动：http://www.amazon.cn/b?ie=UTF8&node=292588071。——编者注

自己的团队，而且代表整个公司行事。他们绝不会说“那不是我的工作”。

创新与简化

领导者期望并要求自己的团队进行创新和发明，并始终寻求使工作简化的方法。他们了解外界动态，从各处寻找新的创意，并且不局限于“非我发明”的观念。当我们开展新事物时，我们要接受被长期误解的可能。

决策正确

领导者在大多数情况下都能做出正确的决定。他们拥有卓越的业务判断能力和敏锐的直觉。

选贤育能

领导者不断提升招聘和晋升员工的标准。他们表彰杰出的人才，并乐于在组织中通过轮岗磨砺他们。青出于蓝，冰源于水，领导者培养的也是领导者，而且他们严肃地对待自己育才树人的职责。

坚持最高标准

领导者有着近乎严苛的标准——这些标准在很多人看来可能高得不可理喻。领导者不断提高标准，激励自己的团队提供优质产品、服务和流程。领导者会确保任何问题不会蔓延，及时彻底解决问题并确保问题不再出现。

远见卓识

局限性思考只能带来局限性的结果。领导者大胆提出并阐明大局策略，由此激发良好的成果。他们从不同角度考虑问题，并广泛寻找服务客户的方式。

崇尚行动

速度对业务影响至关重要。很多决策和行动都可以改变，因此不需要进行过于广泛的推敲。我们提倡在深思熟虑的前提下进行冒险。

勤俭节约

我们尽量不在与客户无关的地方花钱。勤俭节约可以让我们开动脑筋、自给自足并不断创新。

在人员数量、预算规模或固定开支方面，没有额外的投入。

自我批评

领导者并不认为自己或其团队总是对的。他们会主动揭短，即使这样做会令自己尴尬或难堪。领导者会以最佳领导者和团队为准绳来要求自己及其团队。领导者在错误中学习，而不是文过饰非。

赢得信任

领导者会真正敞开胸怀、认真倾听，并愿意谦逊地审视自己最坚定的信念。

刨根问底

领导者深入各个环节，随时掌控细节，并经常进行审核。不遗漏任何工作。

敢于谏言　服从大局

领导者必须要能够不卑不亢地质疑他们无法苟同的决策，哪怕这样做让人心烦意乱，精疲力尽。领导者要信念坚定，矢志不移。他们不会为了保持一团和气而屈就妥协。一旦作出决定，他们就会全身心地致力于实现目标。他们愿意支持不受欢迎或难获理解的意见。

达成业绩

领导者会关注其业务的关键决定条件，确保工作质量并及时完成。尽管遭受挫折，领导者依然勇于面对挑战，从不气馁。

“多么希望我在涉足产品经理一职时手边就有这本书。Gayle和Jackie不仅能帮助你获得产品经理的职位，还会教你如何成为一名杰出的产品经理。并且，她们还给出了规划产品经理事业的致胜法宝。”

——Ken Norton，谷歌风投合伙人（原谷歌产品经理）

“如果你在寻找一本全面、基于充分研究的产品经理面试图书，这本就是你最好的选择。Gayle和Jackie深入分析了成为产品经理的整个过程，融汇了产品经理分享的各种观点，其提供的资源使你在求职期间无需费神就能够好好利用。”

——Jason Shah，原微软Yammer产品经理，Udemy课程“如何成为产品经理”讲师

“Gayle和Jackie详细地介绍了面试的方方面面，包括技能的自我定义、简历、产品问题……这本书是成为产品经理道路上的必备手册。”

——Ritu Jain，PM Fast Track Community组织者，LearningJar公司CEO

“在LinkedIn上收到谷歌招聘人员发来的消息后，我翻开了这本书。当时我正在应聘项目经理一职，并最终获得了成功。在互联网公司拥有一席之地，不仅要靠运气，还要有相关经验，并且通过层层面试。这本书帮了我的大忙。”

——Amazon.com读者评论

“这本书针对的是具有计算机科学背景、有志于成为产品经理的人们。具有工程和技术背景的人都能从中获益（我是一名机械工程师）。这本书的后半部分讲解了简历、求职信、产品问题以及可以应用于其他技术领域的案例；前半部分主要介绍了谷歌、微软、Facebook和亚马逊等公司的招聘流程。”

——Amazon.com读者评论

图灵社区：iTuring.cn
热线：(010)51095186转600
分类建议 计算机/IT人文
人民邮电出版社网址：www.ptpress.com.cn



ISBN 978-7-115-38390-7
定价：59.00元

看完了

如果您对本书内容有疑问，可发邮件至contact@turingbook.com，会有编辑或作译者协助答疑。也可访问图灵社区，参与本书讨论。

如果是有关电子书的建议或问题，请联系专用客服邮箱：ebook@turingbook.com。

在这里可以找到我们：

微博 @图灵教育 : 好书、活动每日播报
微博 @图灵社区 : 电子书和好文章的消息
微博 @图灵新知 : 图灵教育的科普小组
微信 图灵访谈 : ituring_interview，讲述码农精彩人生
微信 图灵教育 : turingbooks